Applicant Name: McLaughlin, Cristina
Title of Project: Interactive Data Visualization for Contact Tracing Statistics
Mentor Name: Prof. Darren Carlson, College of Engineering

COMPLETION REPORT

The Undergraduate Research Opportunities Program allowed me to continue building on a previous UROP project, "Stochastic Simulation of Nosocomial Disease Propagation", conducted by Tamra Oyama and Jie Zhou. Their Spring 2017 project involved programming a motion script in Netlogo, a modelling environment. The model represented infectious disease spread within a health care unit under different conditions and included a Susceptible, Infected, Recovered (SIR) mode. The layout was a scaled version of the John A. Burn's School of Medicine's (JABSOM) SimTiki Simulation Center. Within the simulation, health care workers and visitors follow motion scripts which move them around the center. The motion script was based off data from two observational studies conducted by WOMBAT and Michael Peter Abbey from Griffith University. The Monte Carlo simulations were conducted with different probabilities of transmission and effective distances of infection. The purpose of my project was to take the raw data created in these logs and extrapolate it into a visual model. My main desire was to focus on interactivity of the data. This completion report will discuss my research findings, accomplishments, and experiences I had along the way.

Research and Test Charts Using D3.js

Prior to this project, I did not have experience in application programming, JavaScript, or the d3.js library. This part of the project was the most difficult and time-consuming because it required many hours of dedicated self-directed learning that did not lead to immediate results. However, I did gain some helpful insights into using d3.js by recreating examples online and following tutorials. During my research I ran into multiple issues with the library's versions; often there were useful charts available, but the code was out of date (currently d3 is in Version 4, but many examples use Version 3). After gaining a basic understanding of d3.js, I implemented several graphs using the simulation data including a Gantt chart and force directed graph that were created using basic HTML. Syntactically, d3 implementation was unlike any code I had written before; there were multiple ways to create a single graph using different functions available in the library. After implementing some versions of data manipulation, I discovered that graphs were unexpectedly small—despite the how large the dataset appeared the simulation only produced information for ten agents.

Scaling Up the Simulation

To create more impressive charts, I decided to expand the simulation up to a hospital wing rather than using a small health care unit. This scaled the dataset up to 70 moving agents, multiple infection transfer surfaces (such as chairs or computers), and more than 20 rooms. The agents included 50 non-moving patients, 8 nurses, 3 hospitalists, 10 visitors, and 3 general workers. The most problematic part of this stage was hard-coding motion scripts for 25 moving agents. Each script was about 100 lines of movement code that required specifying each waypoint an agent had to move to. For reference, the floorplan contained about 200 waypoints.

I also reproduced practices in Jie and Tamra's original project during the running phase. Their project ran each log for a program equivalent of 48 hours and was completed 20 times. After recreating these runs, I noticed that the upscaled version of the logs took much longer to process than the original version. This was due to massive amount of buffering between each movement set; the SIR model had to account for a possible 70^2 or 4900 contacts compared to the original 10^2 or 100. While this computer simulation is an efficient and ethical way of collecting data, the NetLogo modeling program has displayed an inability to effectively handle large amounts of moving agents. If this is expanded upon in the future to accurately model an entire hospital, the simulation platform will need to change.

Processing Algorithm

The next step was implementing an algorithm to process the .json files of the upscaled simulation; I decided to modify a version of Dijkstra's Shorted Path algorithm. Dijkstra's Algorithm finds the shortest path between nodes in a network graph. The pseudocode for the processing algorithm is as follows. First, a 2-D array is created according to the number of agents, in this case it is a 70 x 70 matrix. Each agent is assigned to a top and side of the matrix. The program cycles through the produced .json file and calculates a weighted sum of the distance and duration of each point. The program updates the matrix cell corresponding to the contact between two individuals; if a contact has already been recorded in that cell, the cell is updated with an average of the two values. The values in the matrix were referred to as the "probability of infection". Finally, to produce a list of super spreaders, the program looks for 1) the agent with the most columns filled, 2) the agent with the highest infection to contact ratio, and 3) the agent with the highest infection average.

Application Implementation

The last step in the project was implementing the original charts on an application interface. I immediately ran into a major problem when starting on this step; I chose to use React and Meteor as the basis of my application design. However, the original HTML files with the example chart code were not easily transferable onto the new platform. React is a view lifecycle management system that's popular for application framework; its major feature is keeping track of a copy of the document object model (DOM), and only renders elements that need to change. The main challenge was that d3 also wants to control the DOM, so I needed to create functions in different React lifecycles to make sure that the visualization updated on different mouse events. Unlike the beginning of my project where I could easily find examples of HTML d3 examples, there were very little helpful examples of integrating d3 with React on the Meteor platform. I found a single example of the d3-React lifecycle and expanded that code to apply to each of the graphs that were implemented. The main visualization was a force-directed graph. In the graph there are lines between each node (agent) based on the contact average described in the processing algorithm. The heavier the contact average, the thicker and stronger the link is. By default, this means the social butterfly, super spreader, or person with most contacts gravitates towards the middle of the graph. I was able to verify this by the comparing the center node to the printed output of the matrix algorithm. Other graphs included heatmaps of agent contact, and cluster graphs of "agent cliques".

Overall, I believe the project was a success. The final product is a tool that will be useful in analyzing future test results produced by next steps in this expansive contact tracing project. The results show that we can model infection spread and implement useful graphs to visualize the salient information. Through this research I learned a lot about myself and what I'm interested in. I found out that I enjoy implementation much more than basic research, and that I require tangible results to stay interested. After learning about this in the first portion of my project, I started setting daily goals for myself to stay on task and feel accomplished. I also met with my advisor bi-weekly to discuss progress and focus on points to expand on. Lastly, the project also taught me about setting realistic goals, there were some parts of the project that were impossible to implement with my current skill set.

Experience as a UROP Recipient

I think experience has been rewarding in that it pushed me out of my comfort zone. I probably would not have applied without the support of Tamra, Jie, and my advisor, Dr. Carlson. It was daunting to work alone on a project of this scale and I regret not applying as a team with other people. UROP allowed me to do research on topics I otherwise would not have and gave me a chance to present what I learned at the end of the semester. I did have some issues with receiving the funding and did not originally know that the stipend would be applied to my parents' loan for tuition. I think this could have been made clearer in some of the presentations given by UROP to incoming applicants. Overall, my experience has been positive, and I think it is a great opportunity afforded to undergraduate students.