# Parallel Epidemic Simulation


| Mathew Reilly | Adrian Hernandez | Anthony Cobb | Sebastien Joseph | Tra'Von Ross |
|---|---|---|---|---|
| ma288914@ucf.edu | aahernandez@ucf.edu | an824794@ucf.edu | se788033@ucf.edu | tr500226@ucf.edu |


## Challenges:

Considerations we made to resolve implementation challenges:

- We are using the SIR model where each 'living' cell can be in a state of susceptible, infectious, or removed. Susceptible is when they are able to become infectious, and a cell may become infectious if another cell that is already infectious comes near a susceptible cell. A cell may go into a removed state, meaning that they are no longer infectious and are immune to the infection, when the cell has been in the infectious state for a certain amount of time.
- We decided to go with a grid based approach to make multithreading a single simulation easier.
- We wanted to make one simulation take advantage of multithreading so we split our grid subgrids where each thread can be in charge of their own section of the grid. To handle the interaction between the sections of the grid, the program waits until all threads finish working on the internal part of each subgrid (everything in the subgrid except the borders).

Some roadblocks we experienced:

- How we wanted to structure our data. (grid based or uniform space)
- How we wanted to multithread the simulation. (split the grid space or multiple simulations at the same time)
- Planning out the parallelization. Making sure that the parallelization took into account that the simulation is done on a single grid, making sure that threads don't overwrite work done by other threads.
- What kind of statistics we wanted to gather.

Work left to be done:

- Statistic collection
- Simulation rendering
- Multithreading

Creating a design that is reflective of the SIR model while also being possible to implement.

# Tasks:

Completed Tasks:

So far, the tasks we have completed include designing requirements, setting up our local environments, creating our initial grid, creating an initial SIR algorithm, and defining how to run the algorithm. The initial task of design requirements was done as a group, this step was of vital importance so that we knew how our project implementation will support our idea for a simulation. It is in this gathering of requirements that we decided to use a grid based system with a SIR type model being reflected in cell types: susceptible, infectious, and removed. Additionally, we decided on ideas on how to parallelize the grid, collect data, and visualize the grid. After our initial requirements were defined, we were able to choose our environment, where we chose to work using Java 21.0.2 and organized our branches. Here we tested and ensured we could have working local environments and could effectively make changes to the repository. Next, the grid implementation was set up, where a 2D array holds Cells. Supporting classes were created and the initial grid was defined. Instantiating the grid and creating a method for printing its current state were also built. From here implementation focusing on our algorithm for infections spreading and removing cells would be created in conjunction with being able to step through each simulation moment.

Remaining Tasks:

Our next steps in completing our project are enforcing the algorithm to run in parallel, collect and process data collected between the steps, and visualizing the grid. Each of these steps will provide our program essential functionality improving its runtime, representing how the data changes, and visualizing the changes seen while it runs. Additionally, once these tasks are completed, we will be able to collect information for our final report and record our presentation.

# Goals:

Why we chose this project:

The reason that this project was chosen was due to the fact that society was caught off guard by the recent pandemic. A plethora of people started getting sick due to the viral disease. Had we possessed a way to simulate the spread of a virus on the population, we could have taken quicker mitigative measures. The virality of the virus could have been assessed much sooner which would have hastened the implementation of events such as the lockdown, testing, and the closing of airports. Even after the pandemic has already occurred, we can still utilize the simulation results to assist in the allocation of resources determined on the geographical spread of the virus. This would relieve the stress on hospitals and health care workers that end up overworked and understaffed.

We want it to be easy to understand:

The multithreaded SIR model is for anyone looking to understand the impact of a simulated epidemic. It aims to supply a surface level representation of the epidemics that can showcase the core aspects of the SIR model without being too computation heavy.

Provide control to users:

The idea of this version of the SIR model allows users to run the simulation more quickly using multithreading. Users would be able to run multiple simulations, using different parameters each time, to predict and understand the impact of an epidemic. There are two types of parameters: necessary and additive. The necessary parameters include population size, infection rate, and removal rate (how long cells stay infected). These parameters are necessary for the model to run. The additive parameter is the visitation rate (how likely it is for a cell to visit another community). This parameter is not necessary, but adds to the variability of the simulation.