VIRUS TRANSMISSION SIMULATION
FINAL PROJECT REPORT

PROGRAM STRUCTURES AND ALGORITHMS (INFO6205)
SPRING 2021

Professor: Robin Hillyard

Submitted By:

Charmi Dalal - NUID: 001582441

Apeksha Khandelwal – NUID: 001091651

Anusha Chandrakanth Lingadahalli - NUID: 001007696

1. Introduction

The scope and scale of the COVID-19 epidemic are unprecedented in modern times. As a novel coronavirus with a high rate of propagation, its emergence caught the world unprepared, with little or no natural immunity amongst the population, and no existing effective vaccine. The impact has been staggering, in both social and economic terms. The global health toll to date is more than 140 million confirmed cases and more than 3 million deaths, exhausting healthcare systems and presenting significant concern about long-term health impact even for those who experienced mild cases. Initial attempts to control the spread have damaged national economies and disrupted global trade, thus exposing the vulnerabilities of existing systems, and serving to erode public confidence in and adherence to government efforts to bring the situation under control.

The main first-line methods used by governments to suppress the spread and mitigate the impact consisted primarily of social distancing, the use of masks, the restriction or lockdown of non-essential businesses and quarantine.

The recent emergence of vaccines is a welcome development, as it introduced more effective pharmaceutical responses to the policy mix. The vaccine-based approach is expected to become the dominant strategy, gradually diminishing the reliance on the NPIs and their associated disruptions. Vaccines such as Pfizer-BioNTech COVID-19, Moderna COVID-19, Sputnik V completed their clinical studies and received regulatory approval for public use in many countries. Initial results indicate high efficacy rates for these vaccines, usually above 90%, after the two-shot individual regimen is administered.

The effective large-scale deployment of the vaccines can significantly increase the percentage of the population exhibiting a level of immunity to the virus. Combined with those previously infected, and thus possessing a degree of natural immunity, it now becomes feasible for communities to achieve levels of immunity sufficient to be described as “herd immunity”, which is considered the surest way to suppress the epidemic, ongoing, and to protect the most vulnerable groups of the society.

With the cases still surging and showing different behaviors in different areas, researchers are turning to mathematical models to predict how the disease is spreading. These models consider different factors related to the virus along with specific population factors and simulate how the virus will behave. Our application is a Virus Transmission Simulator capable of visualizing the spread of a viral disease and its impacts, by providing real-time graphs and statistics to better understand the simulation, taking various factors into account such as quarantine/social distancing, R factor of the disease, usage and effectiveness of masks, prevalence of testing and contact tracing and availability and efficacy of vaccines.

2. Aim of The Project

The aim of this project is to simulate and compare transmissibility and mortality rates for severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) with those of other epidemic coronaviruses, such as pandemic influenza viruses and understand how factors like the K value, R value, population density, infection range, mortality rate and various non-pharmaceutical interventions like social distancing, mask mandates, and travel restrictions impact the outcome of the viral outbreak. Our goal is also to provide a GUI to facilitate the user to interact with the application easily and provide an option to tweak various factors, simulate how the virus behaves and view the results. It has been enhanced by detailed outputs consisting of graphs and charts to analyze the simulation data and the impact of policies on virus transmission.

3. Project Details

Virus Transmission Simulator is a stochastic simulation model that can depict the spread of different viral diseases. This model works by considering the R and K factors of the disease, the mortality rate, recovery time and projecting the number of infections and state of the outbreak based on the healthcare capacity, intervention, and protection policies.

Technology Used

The simulation is developed using Java 8 with AWT & Swing APIs to create GUI. Graphics are rendered using Graphics2D library which provides sophisticated control over geometry, coordinate transformation, color management and text layout. Unit testing is implemented using JUnit framework for Test Driven Development. GitHub is used as VCS & its Actions workflow for continuous integration of project.

Simulation Details

The program provides user the functionality to simulate any viral disease by allowing the user to configure different parameters such as R value, K value specific to the disease, mask risk rate, vaccination rate, quarantine time and so on in the config.ini file.

- Simulation is initialized with total population and some unhealthy population, represented by the balls.
- States of people are color coded in balls as follows:

  - Yellow: normal people
  - Red: positive patients
  - Orange: suspected patients
  - Black: dead patients

- Balls move randomly within the boundary of space and healthy ball can be infected by colliding with infected ball, which depends on rate of infection.
- Infected balls will be recovered after certain time and cannot be infected again. Any ball can be marked for social distancing or quarantine, which means it cannot move during the simulation, but other balls can collide with it. Number of social distancing balls can be set from the panel.
- If any ball is marked as vaccinated, then its efficacy is determined along with fatality rate.

4. Implementation

During the virus transmission, the social contacts between susceptible and infected persons are significant, whereas contacts between uninfected persons are not significant. The overview of this simulation is building a contact network as people become infected at each simulation step by generating only infected persons and their close contacts. The main advantage of this simulation is computational efficiency when simulating disease outbreak with high prevalence in a large population and the impact of vaccination.

To create a sufficiently competent model, we start with initializing various statistics about the virus and other user provided data, a set of random individuals are initialized which will contain various variables using which we start the simulation.

Some of the following statistics are being used for simulation:

1. R Value: R value or the reproduction number is defined as the average number of individuals an infected person will infect over the course of the spread.
2. K Value: K value or the dispersion value is defined as the variance in the number of individuals an infected person will infect.
3. Safe distance: The physical distance which is safe to prevent transmission.
4. Mask risk rate: This shows the mask effectiveness.
5. Reinfection risk: The risk of having reinfection of virus (post vaccination)
6. Quarantine time: The time of quarantining.
7. Vaccination discovery time: The time for vaccination known.
8. Death variance: Used to determine the death of suspected patient.

How R and K values are being incorporated in the model:

R Factor: R-naught is the rate of transmission and is the average number of people infected by one infectious individual in a population with everyone susceptible. Higher values indicate greater infectivity.

$$Ro = ßï$$

Ro = basic reproduction number,
ß = infection producing contacts per unit time, &
Ï = mean infectious period

Some typical Ro values:

• COVID-19: 2.0 – 4.0

K-factor: It explains the dispersion of the infection and can be calculated as product of the rates of distribution and infection related to virus.

$$K = Distribution * infection$$

Distribution= measures the average number of people a host will contact while still infectious, &

Infection= measures how likely an average person also becomes infected after contact with a viral host

Virus having a k-factor of 1 is in a "steady" state of neither growth nor decline, while a k-factor greater than 1 indicates exponential growth and a k-factor less than 1 indicates exponential decline.

Some typical K values:

• COVID-19: 0.1-0.3

Simulation Implementation Details:

The following depicts and explains the algorithm developed which models the virus spread and simulates it.

Initialization Phase of the model:

1) Step 1: Load the following virus statistics and user specific data in config file:

    a) R Value K Value

b) safe distance
c) mask risk rate and with mask risk rate
d) contact intention
e) vaccination rate
f) isolation risk rate
g) hospitalized day
h) reinfection risk rate
i) vaccination discovery time
j) quarantine time

2) Step 2: Fetch the values from config file and create the graph and line charts

```
private static void initGraph() throws IOException {
    //reading the config file to fetch the value of variables
    Ini ini = new Ini(new File( pathname: "./src/config.ini"));
    //connecting the file to map for ease in reading and fetching
    Map<String, String> map = ini.get("default");

    //creating the main frame and panel
    JFrame frame = new JFrame();
    JPanel ui = new JPanel();
    ui.setLayout(new GridLayout( rows: 1, cols: 3));

    //calling the class which creates the graph and passing it to thread to ensure continuous movement
    Simulator graph = new Simulator();
    Thread panelThread = new Thread(graph);

    //adding the graph and line charts to main panel and setting the background color
    ui.add(graph);
    ui.add(new Charts());
    ui.setBackground(Color.BLACK);

    //adding the ui to frame and adjusting its attribute
    frame.add(ui);
    frame.setVisible(true);
    frame.setTitle(map.get("virus_simulation")+" of "+map.get("city_population")+" Residents of Boston");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize( width: 1200,  height: 600);
    frame.setResizable(false);
    frame.setLocation( x: 50,  y: 50);

    //starting the thread
    panelThread.start();
}
```

Step 1:Taking resident list (say 5000) get the current health status of the residents.

Step 2: Move the dots over the frame and randomly infect resident and assigns random viral load.

```java
/* randomly infects resident and assigns random viral load */
public void infectResident() {
    float randomLoad = new Random().nextFloat();
    virus.setViralLoad();
    int viralLoad = virus.getViralLoad();
    if (randomLoad < Float.parseFloat(map.get("K")) || viralLoad < Integer.parseInt(map.get("viral_safe_threshold"))) {
        this.isSuperSpreader = true;
    }
    this.infection_status = Integer.parseInt(resident_status.get("suspected"));
    this.infecteDay = Simulator.pandemicDay;
}
```

Step 3: Calculate distance to check if residents are following social distancing

Step 4: Move randomly to spread virus and infect residents

a) Infect the non-infected based on randomly generated chances using a continuous uniform distribution and their susceptibility.
b) If the resident infection status is positive check for pandemic day and possible death day.
c) Calculate and get the quarantine time and check if resident is vaccinated or not

Step 5: check status of patients by counting isolation and mortality rates. And check if quarantine time is over and if yes then does random movement

```java
/* checks status of patiets by counting isolation and mortality rates*/
public void checkHealth() {
    double targetX = MathUtil.stdGaussian( sigma: 100, xAxis);
    double targetY = MathUtil.stdGaussian( sigma: 100, yAxis);
    randomMove = new RandomMove((int) targetX, (int) targetY);
    if (!this.isIsolated)
        doRandomMove();
    /* checks if quarantine time is over and if yes then does random movement */
    if (this.isIsolated) {
        if (Simulator.pandemicDay - this.isolationDay > Integer.parseInt(map.get("quarantine_time"))) {
            this.isIsolated = false;
            doRandomMove();
            Random random = new Random();
            int x = (int) (500 * random.nextGaussian() + city.getCenterX());
            int y = (int) (500 * random.nextGaussian() + city.getCenterX());
            if (x > 500) x = 200;
            reLocate(x,y);
        }
    }
```

Step 6: Check if the patient is recovered and sets infection status to negative and checks for fatality by age

```
/*Checks if patient is recovered and sets it negative*/
if (infection_status == Integer.parseInt(resident_status.get("positive")) && this.possibleDeathDay == 0) {
    if (Simulator.pandemicDay - this.diagnoseDay >= Integer.parseInt(map.get("hospitalized_days"))) {
        float ranDeathPoss = new Random().nextFloat();
        if (ranDeathPoss <= Float.parseFloat(map.get("broad_risk_rate")) / Float.parseFloat(map.get("R"))) {
            this.infection_status = Integer.parseInt(resident_status.get("negative"));
            this.isSuperSpreader = false;
            this.isCured = true;
            this.possibleDeathDay = 0;
            this.isIsolated = false;
        }
    } else {
        int mortalityPoss = new Random().nextInt( bound: 10000) + 1;
        float fatalityRate = virus.getFatalityRateByAge( resident: this);
        if (1 <= mortalityPoss && mortalityPoss <= (int) (fatalityRate * 10000)) {
            int dieTime = (int) MathUtil.stdGaussian(Double.parseDouble(map.get("death_variance")), Integer.parseInt(map.get("deat
            this.possibleDeathDay = this.diagnoseDay + dieTime;
        } else {
            this.possibleDeathDay = 0;
        }
    }
}
```

Step 7: Check if certain days are passed and if not recovered, marks it as dead when infection status is positive.

Step 8: Check risk of residents to get infected with respect to its contact with other residents infection status is negative or isvaccinated or with safe distancing

```
if (this.infection_status < Integer.parseInt(this.resident_status.get("suspected"))) {
    List<Resident> people = ResidentDirectory.getInstance().residentList;
    for (Resident resident : people) {
        if (resident.getInfection_status() == Integer.parseInt(resident_status.get("negative")) ||
                resident.getInfection_status() == Integer.parseInt(resident_status.get("dead")) ||
                resident.getIsIsolating() || resident.isVaccinated() == true)
            continue;
        float ranPossiblity = new Random().nextFloat();
        float risk = virus.calculateInfectionRisk(resident);
        if (ranPossiblity < risk && calculateDistance(resident) < Float.parseFloat(map.get("safe_distance"))) {
            this.infectResident();
            break;
        }
    }
```

Step 9: Decide outcome:

a) Find infected individuals who have been infected for over the specified time.
b) Using randomly generated chances using a continuous uniform distribution and their
c) mortality chances decide they survive for not.
d) Check if the vaccinated resident is reinfected or not

e) Check If the number of currently infected people is extremely above the healthcare capacity at the current moment the chances of death and check for death rate
f) Check for super spreaders, suspected patients and vaccinated people.
g) Check for dead and cured patients and also isolated patients

Step 10: Update Values for Random Individuals and virus statistics and repeat the process for various scenarios.

5. Output

From the configuration file, we can set following parameters:

1. Population size
2. Randomly infected
3. Vaccination rate
4. Quarantine time
5. The R value for the virus
6. The K value for the virus
7. Reinfection risk
8. Hospitalized days
9. Mask effectiveness of 3 types of masks:
   a) Cloth masks
   b) Surgical masks
   c) N95 masks
10. Mortality rates based on age groups:
    a) 0-9
    b) 10-19
    c) 20-29
    d) 30-39
    e) 40-49
    f) 50-59
    g) 60-69
    h) 70-79
    i) 80-89
    j) 90-99

With the following default values in config file:

```
virus_simulation = Spread of SARS-CoV-2 Simulation
randomly_infected = 500
quarantine_time = 140
city_population = 5000
vaccination_rate = 0.01f
fatality_rate = 0.01f
vaccination_discovery_time = 1000
city_area_width = 700
```

```
city_area_height = 800
contact_intention = 0.99f
R=0.7f
K=0.8f
mortality_0_9 = 0f
mortality_10_19 = 0f
mortality_20_29 = 0f
mortality_30_39 = 0.01f
mortality_40_49 = 0.01f
mortality_50_59 = 0.01f
mortality_60_69 = 0.01f
mortality_70_79 = 0.1f
mortality_80_89 = 0.13f
mortality_90_99 = 0.2f
cloth_mask_risk_rate = 0.6f
surgical_mask_risk_rate = 0.4f
n95_mask_risk_rate = 0.2f
hospitalized_days= 280
broad_risk_rate = 0.8f
re_infection_risk=0.8f
death_period = 100
non_isolating_rate = 0.2f
death_variance = 1
mask_wearing_risk = 0.2f
safe_distance = 1.8f
viral_load_threshold =35
viral_safe_threshold =25
[resident_status]
negative = 0
suspected = 1
positive = 2
dead = 3
```

Following is the screenshot which depicts the GUI for simulation of above set values.

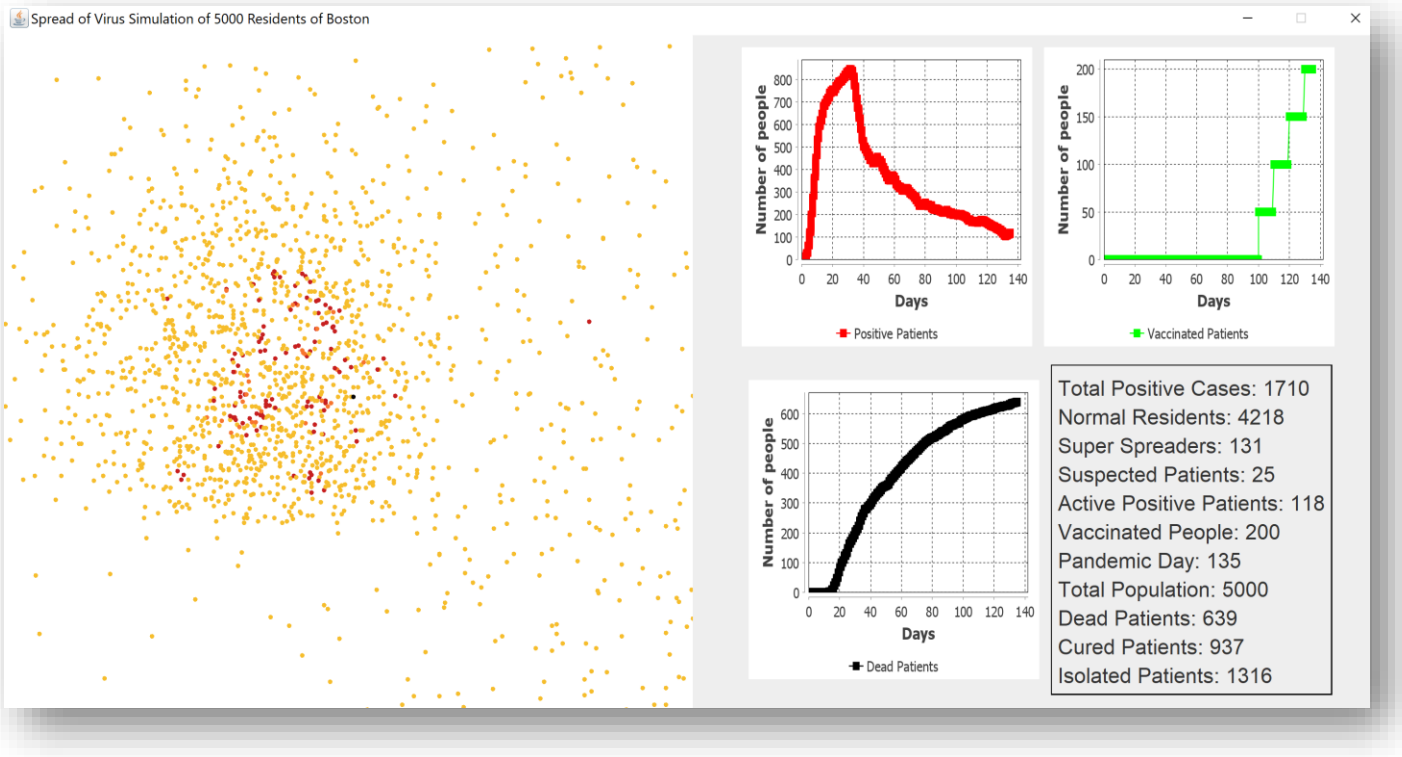


Fig1. GUI of Simulation

Comparison of Covid-19 and Influenza

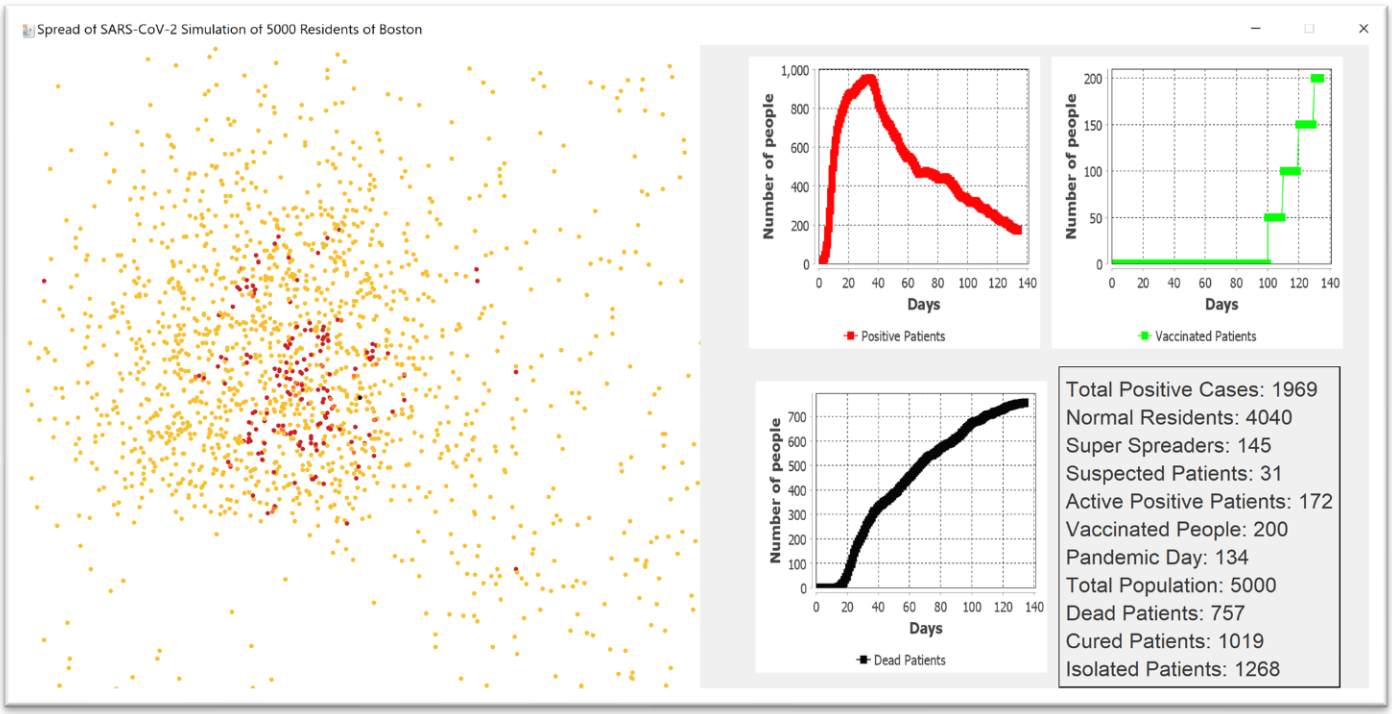
In our application, we have pre-loaded data for Influenza and COVID-19.These two viral diseases have very different characteristics. The K and R values of each of them are given below.

Viral Disease	R value	K value
COVID-19	2.5	0.1
Influenza	1.8	0.9

For covid-19 following values are set for simulation:

```
[default]
virus_simulation = Spread of SARS-CoV-2 Simulation
randomly_infected = 500
quarantine_time = 140
city_population = 5000
vaccination_rate = 0.01f
fatality_rate = 0.01f
vaccination_discovery_time = 1000
city_area_width = 700
city_area_height = 800
contact_intention = 0.99f
R=2.5f
K=0.1f
mortality_0_9 = 0f
mortality_10_19 = 0f
mortality_20_29 = 0f
mortality_30_39 = 0.01f
mortality_40_49 = 0.01f
mortality_50_59 = 0.01f
mortality_60_69 = 0.01f
mortality_70_79 = 0.1f
mortality_80_89 = 0.13f
mortality_90_99 = 0.2f
cloth_mask_risk_rate = 0.6f
surgical_mask_risk_rate = 0.4f
n95_mask_risk_rate = 0.2f
hospitalized_days= 280
broad_risk_rate = 0.8f
re_infection_risk=0.8f
death_period = 100
non_isolating_rate = 0.2f
death_variance = 1
mask_wearing_risk = 0.2f
safe_distance = 1.8f
viral_load_threshold =35
viral_safe_threshold =25
[resident_status]
negative = 0
suspected = 1
positive = 2
dead = 3
```

Simulation results as follows:



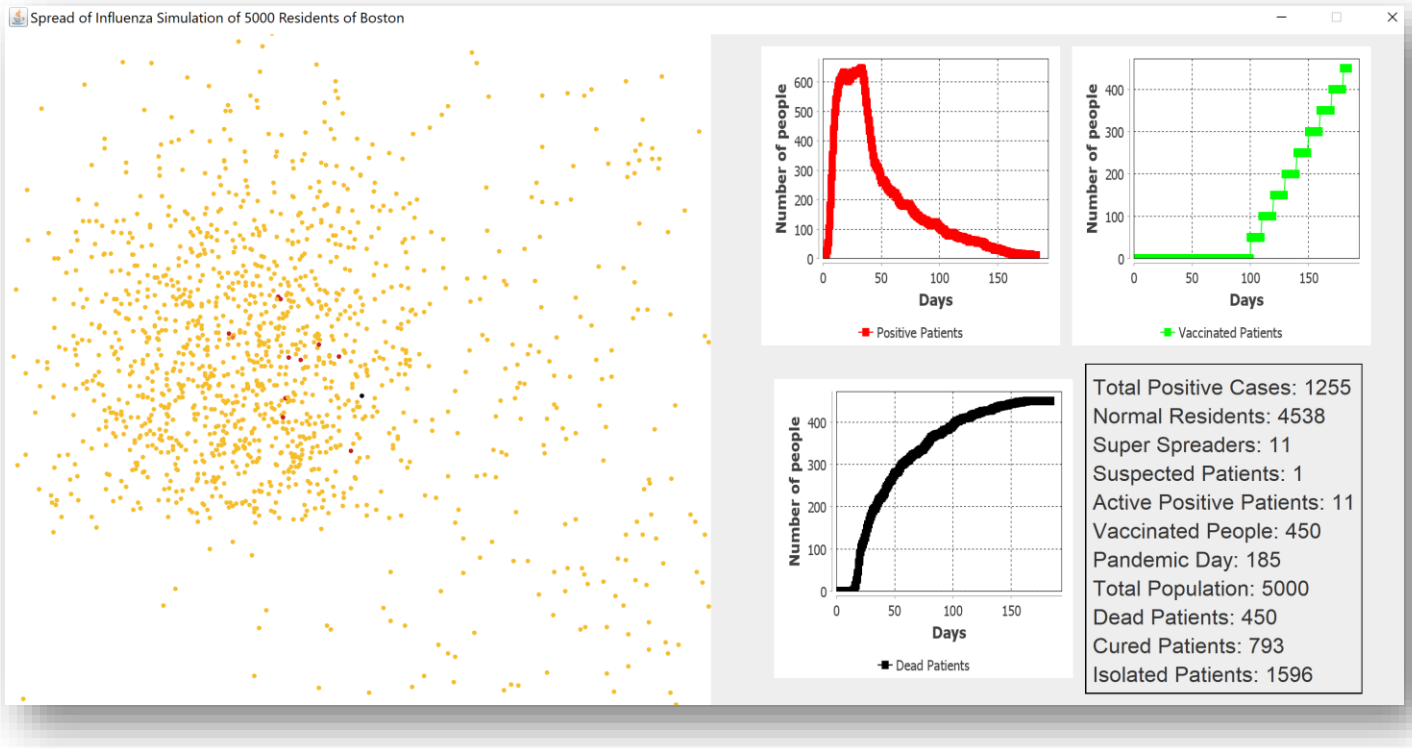
Simulation results for COVID-19

For influenza, the following values are set for simulation:

```
[default]
virus_simulation = Spread of Influenza Simulation
randomly_infected = 500
quarantine_time = 140
city_population = 5000
vaccination_rate = 0.01f
fatality_rate = 0.01f
vaccination_discovery_time = 1000
city_area_width = 700
city_area_height = 800
contact_intention = 0.99f
R=1.8f
K=0.9f
mortality_0_9 = 0f
mortality_10_19 = 0f
mortality_20_29 = 0f
mortality_30_39 = 0.01f
mortality_40_49 = 0.01f
mortality_50_59 = 0.01f
mortality_60_69 = 0.01f
mortality_70_79 = 0.1f
mortality_80_89 = 0.13f
mortality_90_99 = 0.2f
cloth_mask_risk_rate = 0.6f
```

```
surgical_mask_risk_rate = 0.4f
n95_mask_risk_rate = 0.2f
hospitalized_days= 280
broad_risk_rate = 0.8f
re_infection_risk=0.8f
death_period = 100
non_isolating_rate = 0.2f
death_variance = 1
mask_wearing_risk = 0.2f
safe_distance = 1.8f
viral_load_threshold =35
viral_safe_threshold =25
[resident_status]
negative = 0
suspected = 1
positive = 2
dead = 3
```

Simulation results as follows:



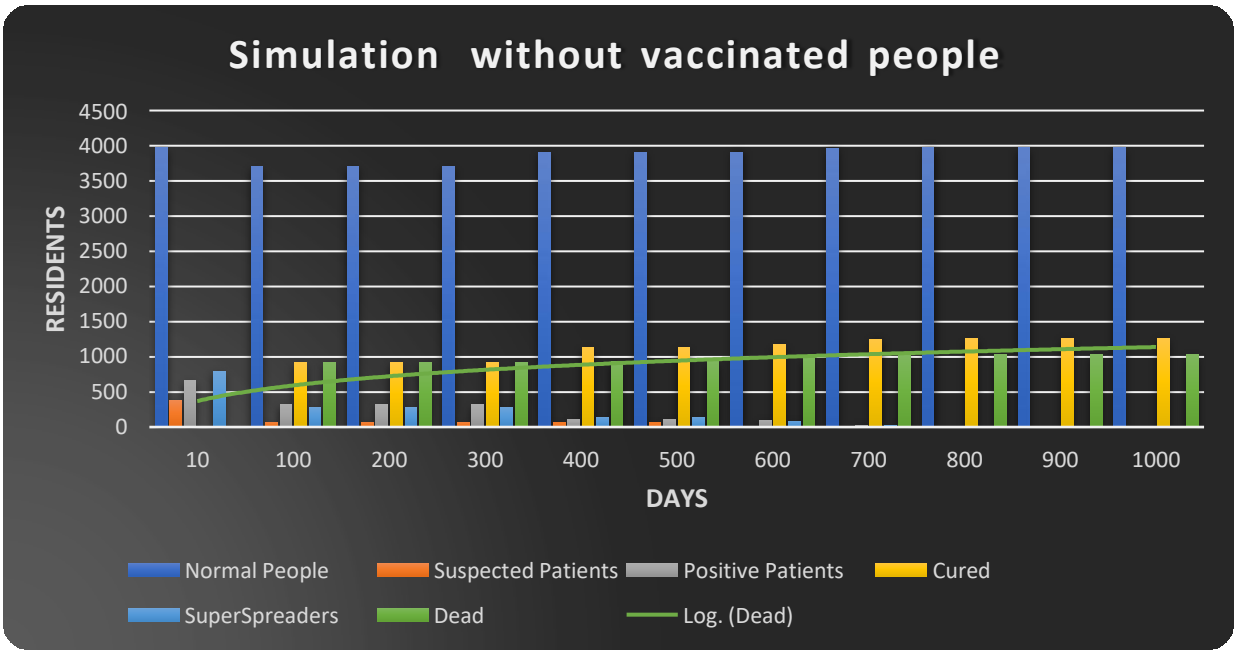
Simulation results for Influenza

The above results of our experiment simulations were as expected. As one can see, in COVID-19 virus (SARS-CoV-2), with a high R value and a low K value, there are more superspreader events and the virus grows rapidly. Also, there is increase in vaccinated people. The entire population gets infected and the death toll 10% or more of the total population or 1097 people.

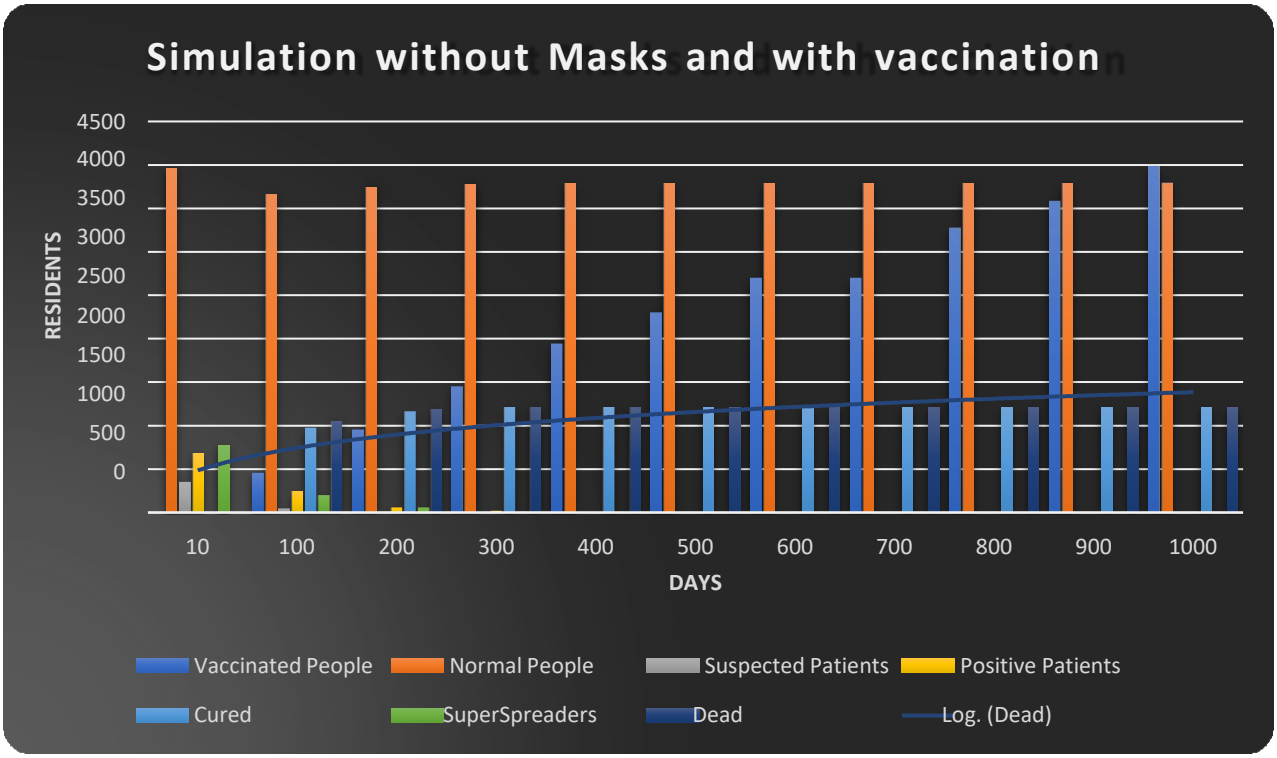
In case of Influenza, with a R value of 1.8 and K value 0.9, the spread of the disease is much slower. Influenza having lower mortality rates brings the death toll to a comparatively very low number of only 631 people. Also, there is increase in vaccinated people.

Graphical representation of various scenarios for Covid-19 transmission

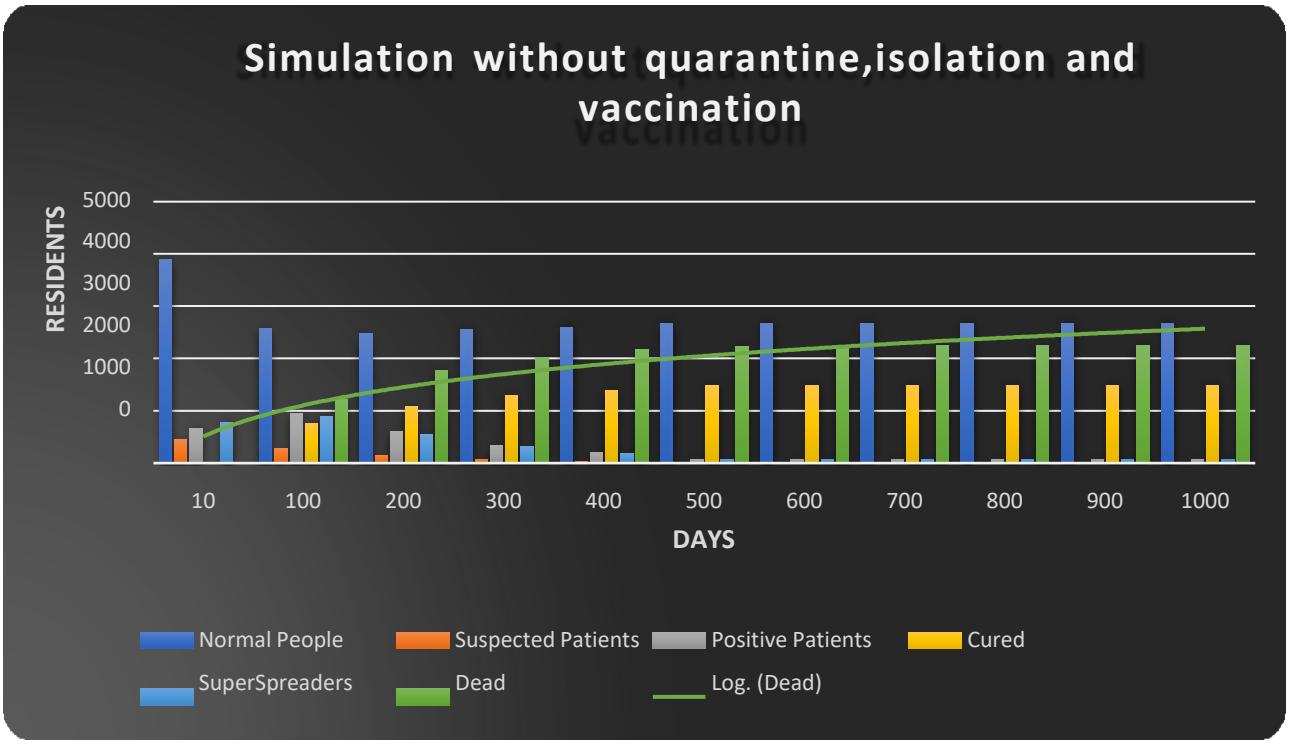
1. Simulation without vaccinated people



2. Simulation without masks and with vaccinated people



3. Simulation – without quarantine, Isolation and vaccination

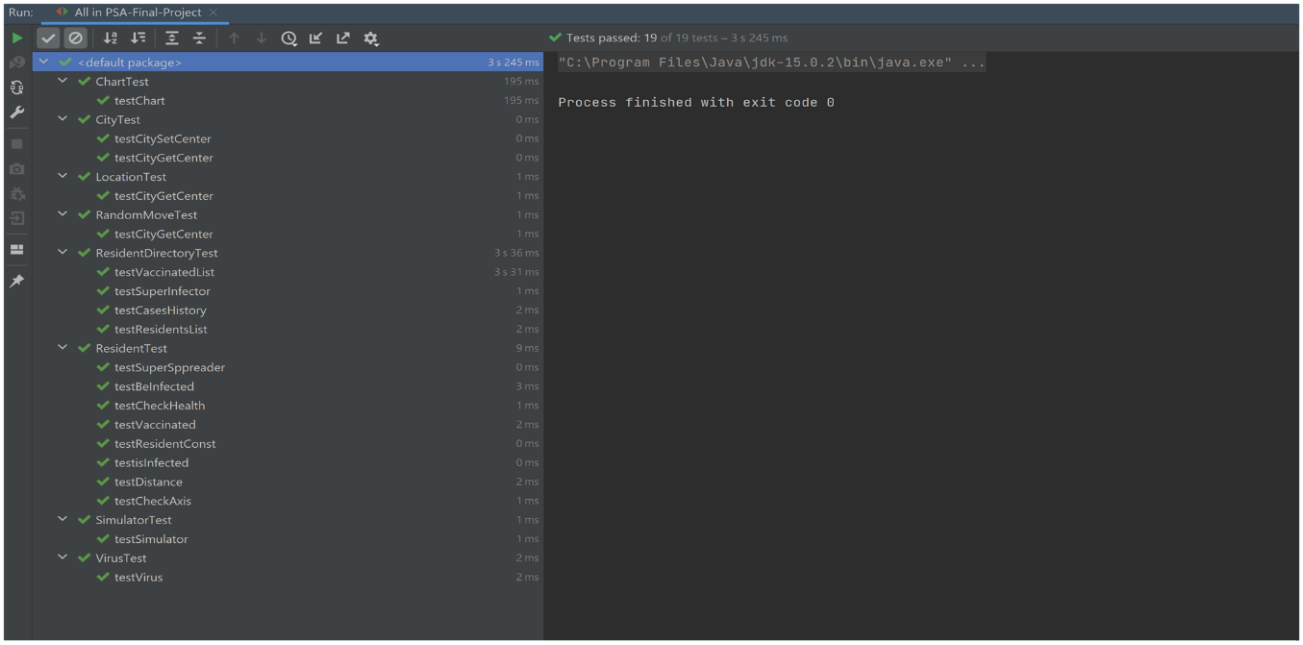


From the above graphs we can estimate the trend in positive patients and dead people for scenarios – without vaccination ,without masks and without quarantine and isolation. When vaccination is not administered there is great rise death rate over period. When masks are not mandated there is higher rise in death rate even though there is rise in vaccinated people. This gives us an overview of how the vaccination can impact in reducing the transmission of virus along with other non-pharmaceutical policies employed effectively.

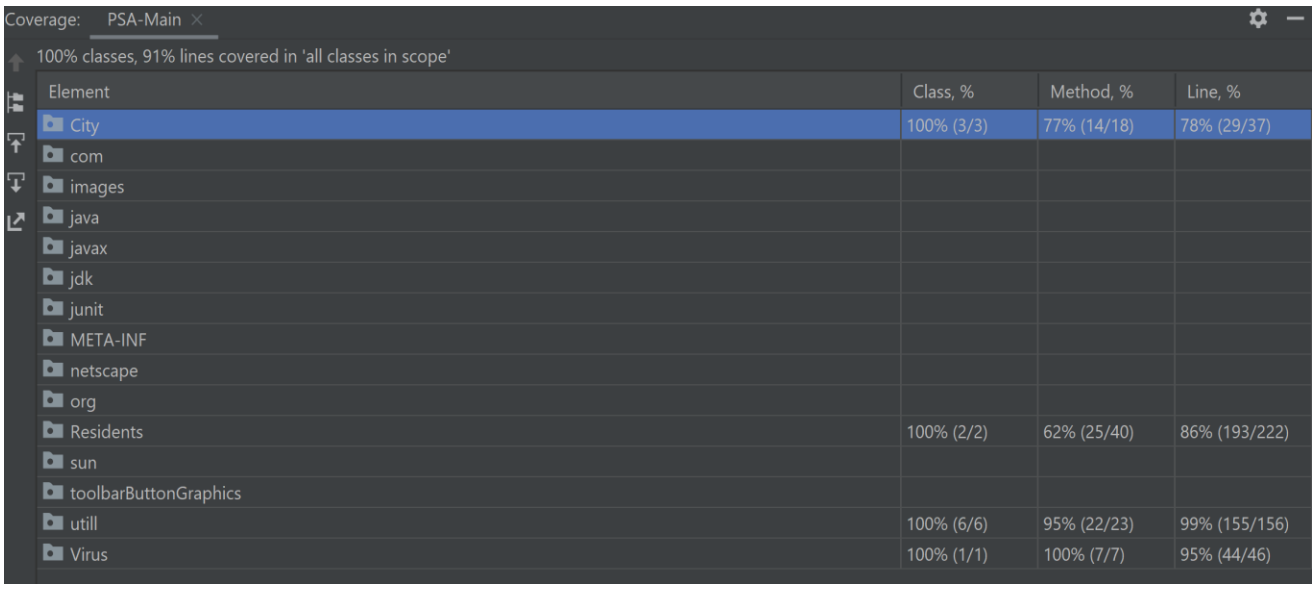
6. Unit Testing

Unit tests have more than 60% coverage by lines.

Following screenshot shows all the tests passed with coverage details:



Screenshot of All tests passed.



Screenshot of full test coverage

7. Conclusion

Simulation model illustrates the interaction between the residents, and the percentage of infectious people present in the population and how vaccination has a positive impact on them. We noticed that the plots show a clear 'tipping point': after 'n' number of infections, the virus spread starts accelerating if we will not apply Mask mandate and Self-Isolation. Increase in vaccinated people results in herd immunity and decreases the infection transmission rate and eventually decreases the death rate. The peak number of infections strongly depends on how many people obey the quarantine, mask mandate and how many people get vaccinated. However, reports have been going around that even without symptoms you can still be contagious, and remain contagious for quite some time after recovering, which makes such a self-isolation scenario and Mask mandate even more important. And, reinfection can occur for those who are vaccinated as well. So the results from the simulation model confirms this in the case of Covid-19.

References

https://www.washingtonpost.com/graphics/2020/world/corona-simulator/

http://gabgoh.github.io/COVID/index.html

https://en.wikipedia.org/wiki/Severe_acute_respiratory_syndrome_coronavirus_2

https://graphics.reuters.com/HEALTH-CORONAVIRUS/HERD%20IMMUNITY%20(EXPLAINER)/gjnvwayydvw/

https://vaxfirst.colorado.edu/

https://www.npr.org/sections/health-shots/2021/02/18/967462483/how-herd-immunity-works-and-what-stands-in-its-way