



Spread of virus

Course: INFO6205

Course Name: Program Structure & Algorithm

Professor: Robin Hillyard

Team: 3

Submitted By:

Xiaoran Li

NUID: 001023070

Ruizhe Zhang

NUID: 001050327

Chenghuan Li

NUID: 001069554

Academic Term: Fall 2020

Date of Submission: Dec 11,2020

Content

1. Introduction about the topic	3
2. Aim of the project	3
3. Complete project details	3
3.1 Simulation Implementation Algorithm	3
3.2 GUI	4
4. Implementation - charts, algorithm	4
4.1 ParaType	4
4.1.1 Country Class	4
4.1.2 Policy Class	5
4.1.3 Virus Class	5
4.1.4 Person Class	5
4.2 Core Simulate Algorithm	6
4.2.1 Simulation Process	6
4.2.2 Generate random number	7
4.2.3 Calculate R value	7
4.2.4 Calculate infectious rate	7
4.2.5 Calculate the total number of infected people	7
4.3 Program UI interface	8
4.3.1 Welcome interface	8
4.3.2 Main interface	8
4.3.3 CustomizeCountry	9
4.3.4 CustomizePolicy	9
4.3.5 CustomizeVirus	10
5. Output	11
5.1 Unit Test	11
5.1 Simulation Output	11
6. Mathematical analysis/evidence	15
6.1 Same country with Different Policy	15
6.2 Different country with Same Policy	16
6.3 Different virus with Same Country and Policy	17
7. Conclusion	19
8. Reference	20

1. Introduction about the topic

The spread of COVID-19 this year has severely affected all aspects of society, and it has also had a huge impact on our normal lives. As the overall fear of the epidemic unfolds, this value-added analysis of COVID-19 incidence data is a worthy effort. In order to better understand the virus and contribute to the prevention of the virus, we hope to simulate the spread of this virus by implementing a JAVA program. In addition, it would be beneficial to better understand the possible impact of various public health interventions on the spread of COVID-19.

However, according to the data given by countries around the world in recent months, differences in policies adopted by countries of world ultimately led to different results of virus transmission, so it is necessary to explore the impact of different policies on virus transmission.

2. Aim of the project

First of all, our goal is to simulate the process of virus transmission within a certain period of time by introducing different virus attributes, national population density, and policy implementation differences. Finally, we need to analyze simulation data to get the impact of policies on virus transmission.

Secondly, we need a good user interface to facilitate the user to enter data, simulate process and view the results.

3. Complete project details

We have adopted a limit to the number of contacts of a single person to generate a N-ary tree of contacts, and implement policy interventions after the virus has spread for a period of time to observe the impact of policies on the control of the virus. This simulation method is based on our own guess, and all algorithms and codes are implemented by our own.

With the initial virus carriers as the root node, daily as the unit, and the average daily contact number as the number of child-nodes, the tree of the contact population with the virus (potential transmission) is generated. According to the customized daily contact transmission rate, the status of all people (health, death, infection, infected but unable to transmit) is updated daily. Data on specific virus characteristics, specific countries (i.e., population density) at a specific time, and under specific policies are collected to generate graphs of population changes over time in different states. Explore the law of virus transmission.

3.1 Simulation Implementation Algorithm

(1) N-ary tree

We set the spread of this virus as an N-ary tree. Set the initial infected person as the root of the N-ary tree, each sub-category is the number of people he contacts each day, and the number of days of simulation is the depth of the N-ary tree. Then, the

spread of the virus is simulated by assigning different status (infected or not) to the sub-categories, and the number of spreaders is affected by R_0 and policy factors.

(2) Recursion:

When generating new nodes, we use a recursive algorithm. This algorithm can quickly generate each node, and then form the entire N-ary tree.

3.2 GUI

The Main Panel is divided into 4 parts: virus, country, policy and control panel.

1.Select Virus:

In order to compare the transmission capabilities of different viruses, we added a ComboBox to select different viruses to join the simulation.

The virus contains four attributes, self-healing rate, death rate, R value and K value. The self-healing rate is the probability that the virus will be cured, and the mortality rate is the probability that the infected person will die. The R value represents the ability of the virus to spread, and the K value represents the ability of an infected person to transmit. We want the user to pass in these parameters from the user interface to provide the basic data for the simulation.

2.Select Country:

In order to compare the spread of viruses in different countries, we added a ComboBox to select different cities to join the simulation. Different countries have different population densities, different population densities represent different daily exposures of residents.

3.Select Policy

We have set up four types of policies to explore the impact of different policies or policy combinations on the spread of the virus in different cities. These four policies are whether to wear masks, whether patients are quarantine, and whether to maintain social distance.

4.Control Panel

We create several Buttons to implement, create new viruses, create new countries, combine different policies. In addition, we also need a simulation button and exit button to implement the program simulation and exit.

4. Implementation - charts, algorithm

4.1 ParaType

4.1.1 Country Class

Density (shown as average contact): On average, how many people a person can contact per day, and how many child nodes are there under a person.

PolicyChangeDay: The day that the policy start. Before the Policy change day, the virus spreads freely without interferences and barriers.

```
public class Country {  
    private String CountryName;  
    private double Density; //Average contact  
    private double PolicyChangeDay; //Policy change day
```

4.1.2 Policy Class

There are three policy that can impact the spread of virus.

ifMaskRequired: If residents are required to wear mask.

ifsocialDistance: If residents are required to keep social distance outside the home.

ifTracingInfectedIndividual: If a person is infected, we can know him/her, and quarantine him/her.

```
public class Policy {  
    public static int count=0;  
    private String PName;  
    private boolean ifMaskRequired;  
    private boolean ifsocialDistance;  
    private boolean ifTracingInfectedIndividual;
```

4.1.3 Virus Class

There are 4 factors can identity a virus.

CureRate: The probability that an infected person will heal by himself every day.

If the patient heals by himself, he will not continue to spread.

DeathRate: The probability that an infected person will die every day. If people die, it doesn't spread.

rFac: The infection rate: it is equal to the average number of people spread by each person. The non-interference propagation rate is called R_0 .

kFac: The probability that each infected person has the ability to infect others.

```
public class Virus {  
    private String VirusName;  
    private double CureRate; // the rate of person self healing  
    private double DeathRate; // the rate of person dead  
    private double kFac; // K value of virus  
    private double rFac; // R value of virus
```

4.1.4 Person Class

isinfected: Used to mark whether the node is infected. 0 means healthy, 1 means infected, 2 means death, 3 means infected but cannot spread.

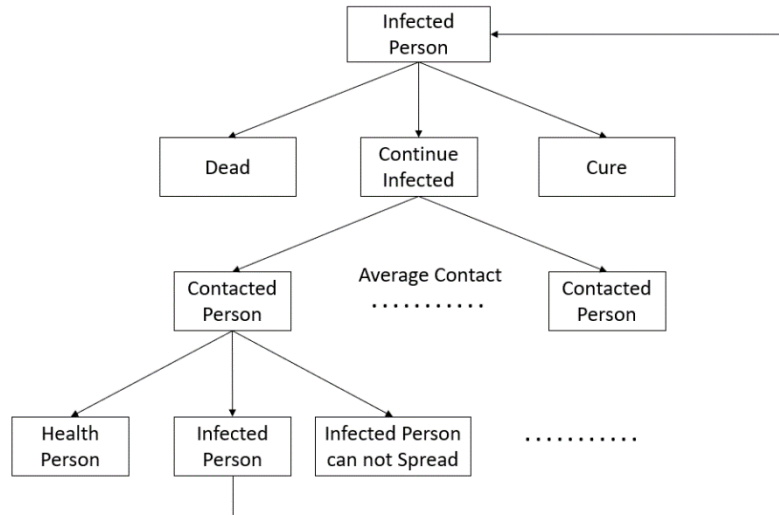
Person []: The person list that everyone comes into contact with every day.

```
public class Person {  
    private int isinfected; //0 - safe, 1 - infected, 2 - dead, 3 - infected but can not spread  
    private Person[] childPerson;  
    private int day;
```

4.2 Core Simulate Algorithm

4.2.1 Simulation Process

Spread process as follows:



p: Initial infected person.

averageContact: The average number of people per person per day

testPeriod: The number of simulate days.

rate: The rate of a person being infected by infected person. (Return by CalculateRate())

deathRate: The rate of an infected person dead.

selfHealingRate: The rate of an infected person becomes health.

kValue: The rate of an infected person has the ability to spread virus.

policyChangeDay: The day that the country begins to use policy. Otherwise, the virus will spread with no effect.

```

public void simulateProcess(Person p, int averageContact, int testPeriod, double rate, double deathRate, double selfHealingRate, double kValue, int policyChangeDay) {
    //to do
    if (p.getDay() < testPeriod) {
        p.setChildPerson(new Person[averageContact+1]);
        if (p.getIsInfected() == 1 || p.getIsInfected() == 3) {
            //judge if person status
            int isDead = getNumber(deathRate);
            int isSelfHealing = getNumber(selfHealingRate);
            if (isDead == 1) { // person dead, stop spread virus
                p.setIsInfected(2); //set person dead status
                this.infectedCount -= 1;
                this.dailyInfected[p.getDay()] -= 1;
            }
            else if (isSelfHealing == 1) { // person self-healing, stop spread virus
                p.setIsInfected(0); //set person dead status
                this.infectedCount -= 1;
                this.dailyInfected[p.getDay()] -= 1;
            }
            else if (p.getIsInfected() == 1) {
                for (int i = 0; i < averageContact + 1; i++) {
                    if (i == 0) {
                        p.getChildPerson()[i] = new Person(p.getIsInfected(), null);
                    }
                    else {
                        int isInfected = getNumber(rate);
                        if (p.getDay() < policyChangeDay) { // the day before using policy
                            isInfected = getNumber(this.noBarriersRate);
                        }

                        int canSpread = getNumber(kValue);
                        if (canSpread == 1) {
                            p.setChildPerson()[i] = new Person(isInfected, null);
                        }

                        if (isInfected == 1 || isInfected == 3) {
                            this.infectedCount += 1;
                            this.dailyInfected[p.getDay()] += 1;
                        }
                    }
                }
            }
        }
        for (Person person : p.getChildPerson()) {
            person.setDay(p.getDay() + 1);
            simulateProcess(person, averageContact, testPeriod, rate, deathRate, selfHealingRate, kValue, policyChangeDay);
        }
    }
}

```

4.2.2 Generate random number

Function `GetNumber()` uses random numbers to implement the possibility of people getting infected under a certain infection rate, that is, a random number (0-1) if this number is lower than the infection rate, they will get infected, otherwise they will remain healthy.

```
public static int getNumber(double rate) {  
    double pr = Math.random();  
    if(pr < rate){  
        return 1;  
    }else{  
        return 0;  
    }  
}
```

4.2.3 Calculate R value

Function `CalculateRValue()` is a function used to calculate R value of viruses.

`totalPerson`: The total number of people who are potential infected.

`dailyIncrease`: The daily increase in the number of people who contacted with the `totalPerson` of the day before.

$R = \text{Total number of infected} * \text{Number of contacts per person per day} / \text{total number}.$

```
public double calculateRValue(int averageContact, int testPeriod) {  
    int totalPerson = 1;  
    int dailyIncrease = averageContact + 1; // include parent  
    for (int i = 0; i < testPeriod; i++) {  
        totalPerson += dailyIncrease;  
        totalPerson -= dailyIncrease / (averageContact + 1); // reduce repeated count person  
        dailyIncrease = dailyIncrease * (averageContact + 1);  
    }  
    return this.infectedCount * averageContact / totalPerson;  
}
```

4.2.4 Calculate infectious rate

Function `CalculateRate()` is used to calculate the probability of each infected person when he contacts with healthy person under the influence of various policies. Its formula is shown in the screenshot.

```
public static double calculateRate(double initRate, double impactOfMasks, double impactOfQuarantine, double socialDistance) {  
    double rate = initRate * (1 - impactOfMasks) * (1 - impactOfQuarantine) * (1 - socialDistance);  
    return rate;  
}
```

4.2.5 Calculate the total number of infected people

Function `CalculateInfectedCount()` is a function that uses simulation process (4.2.1) to calculate the total number of infected people (not including the number of deaths and the number of self-healing).

```

public int calculateInfectedCount(Person p, int averageContact, int testPeriod, double rate,
double deathRate, double selfHealingRate, double kValue, int policyChangeDay) {
    this.infectedCount = 1; //reset
    int[] dailyInfectedTemp = new int [this.testPeriod];
    int simulateTimes = 100; //simulate times
    for (int k = 0; k < simulateTimes; k++) {
        this.dailyInfected = new int [this.testPeriod];
        this.dailytotal = new int [this.testPeriod+1];
        this.dailytotal[0] = 1;
        p = new Person(1, null);
        p.setDay(0);
        simulateProcess(p, averageContact, testPeriod, rate, deathRate, selfHealingRate, kValue, policyChangeDay);

        for(int i=0; i<dailyInfected.length; i++)
        {
            dailyInfectedTemp[i] += this.dailyInfected[i];
        }

        for (int i = 0; i < dailyInfected.length; i++) {
            System.out.println(dailyInfectedTemp[i]);
            this.dailyInfected[i] = dailyInfectedTemp[i] / simulateTimes;
        }

        this.dailytotal = new int [this.testPeriod+1];
        int total=0;
        for (int i = 0; i < dailyInfected.length; i++) {
            dailytotal[i] = total + dailyInfected[i];
            total = dailytotal[i];
        }

        for(int i=0; i<dailytotal.length; i++)
        {
            dailytotal[i]++; //add the initial person
        }

        return this.infectedCount;
    }
}

```

4.3 Program UI interface

4.3.1 Welcome interface



Designer: Xiaoran Li, Chenghuan Li, Ruizhe Zhang

4.3.2 Main interface

We can choose the kinds of viruses or different countries or various combinations of policy. The consequence will be shown below. We can also set which days will the policy public and implement in the table of policy changed day. we can also set the other viruses or countries or policy combinations by the pressing

corresponding button.

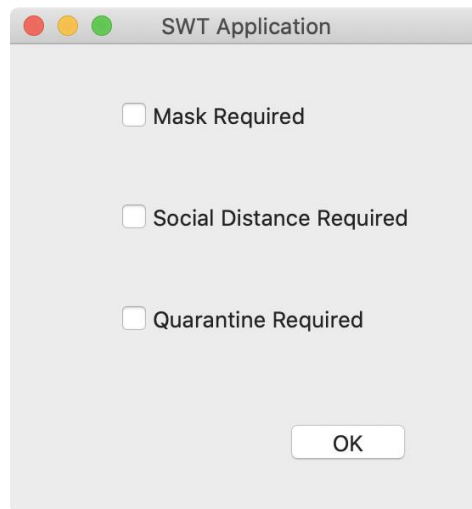
4.3.3 CustomizeCountry

This interface is used to create a new country. There are three parts that can be modified as parameters: 1. Country name 2. Country population density 3. the policy implemented days.

These parameters are set to explore the models of the spread of different viruses in cities with different population densities. The number of days can be modified to predict the number of possible infections.

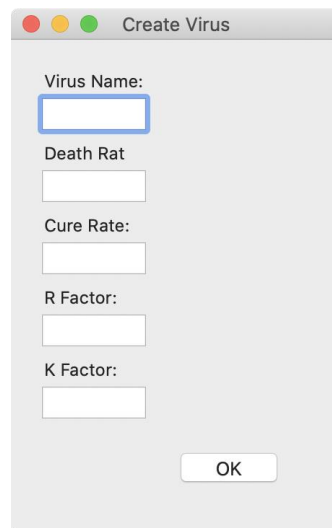
4.3.4 CustomizePolicy

This class is used to implement a new policy combination. There are three types of policies: wearing masks, quarantine, and social distancing policies. We have built-in 2 policies. A strict policy means that all three policies are implemented; open means that all three policies are not implemented. We can freely combine these three policies to form a new combination to detect the impact of these combinations on the spread of the virus in the country.



4.3.5 CustomizeVirus



This class is used to create a new virus. We have built-in COVID-19 and SARS virus, and set its transmission rate and lethality rate. This interface is set up to facilitate us to create a new virus, just enter its name spread rate and lethality rate. This is to explore the spread of different types of viruses under the influence of various policies and urban population density.








5. Output

5.1 Unit Test

Finished after 0.161 seconds

Runs: 4/4  Errors: 0  Failures: 0

▼  FinalProjectTest [Runner: JUnit 5] (0.061 s)

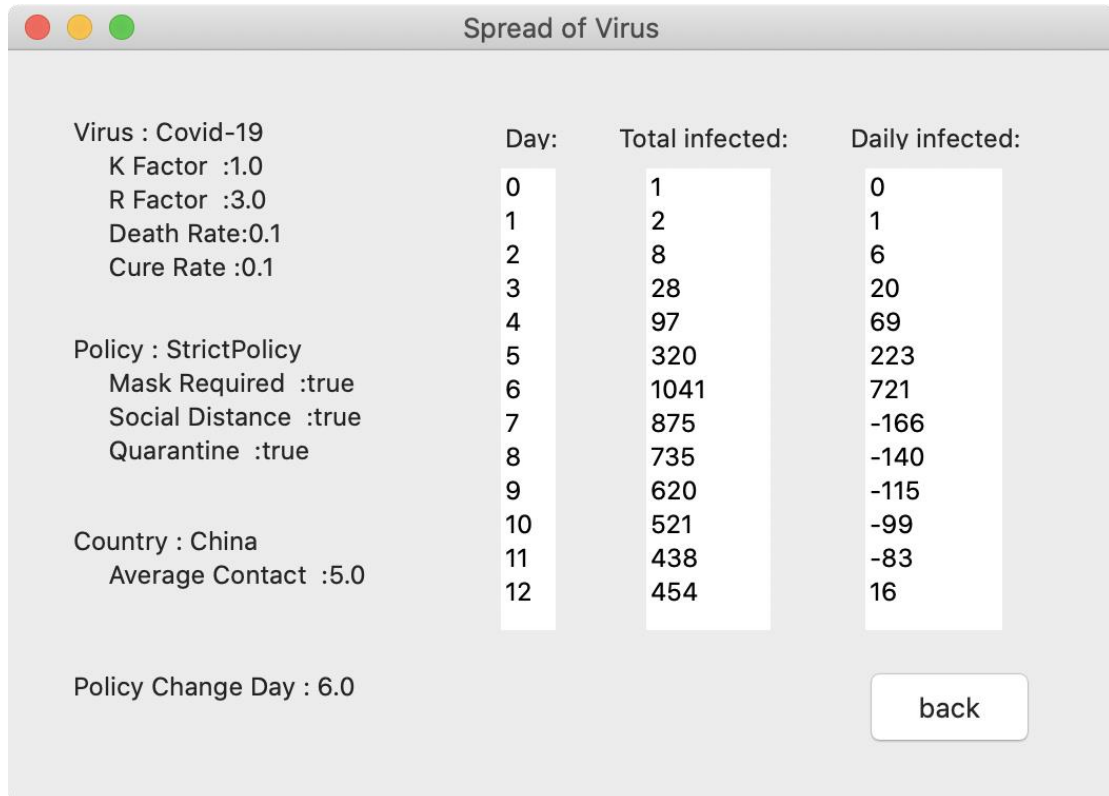
-  testCalculateRValue() (0.027 s)
-  testCalculateInfectedCount() (0.016 s)
-  testCalculateRate() (0.002 s)
-  testGetNumber() (0.016 s)

5.1 Simulation Output

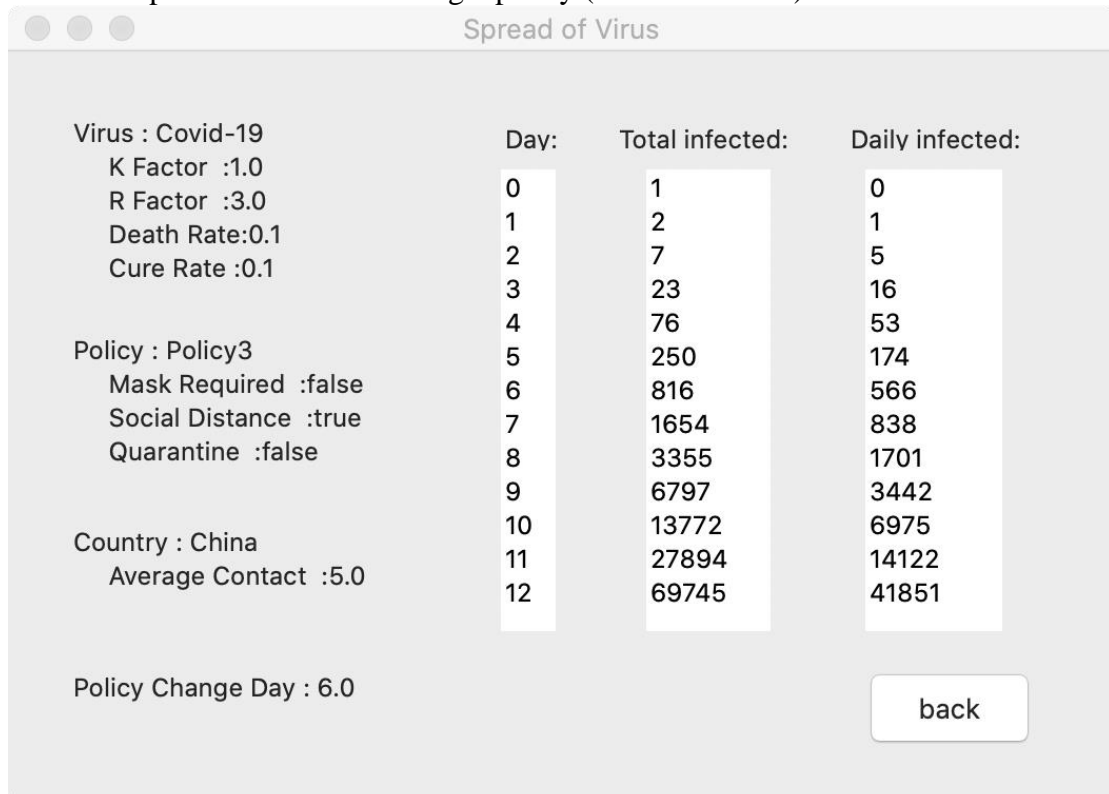
Covid-19 spread in China with OpenPoilcy (no measures spread):

Spread of Virus			
Virus : Covid-19	Day:	Total infected:	Daily infected:
K Factor :1.0	0	1	0
R Factor :3.0	1	2	1
Death Rate:0.1	2	8	6
Cure Rate :0.1	3	28	20
	4	94	66
Policy : OpenPolicy	5	308	214
Mask Required :false	6	1006	698
Social Distance :false	7	3264	2258
Quarantine :false	8	10580	7316
	9	34279	23699
Country : China	10	111061	76782
Average Contact :5.0	11	359819	248758
	12	1439258	1079439
Policy Change Day : 12.0			
			back

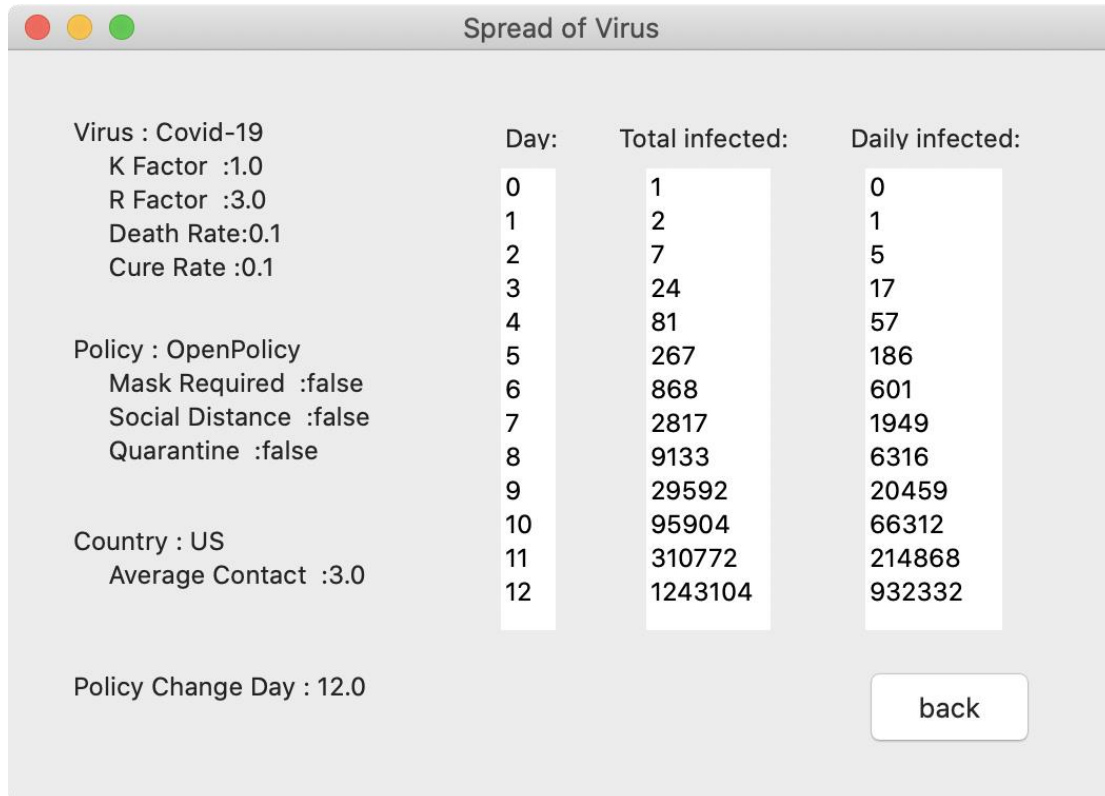
Covid-19 spread in China with StrictPoilcy:



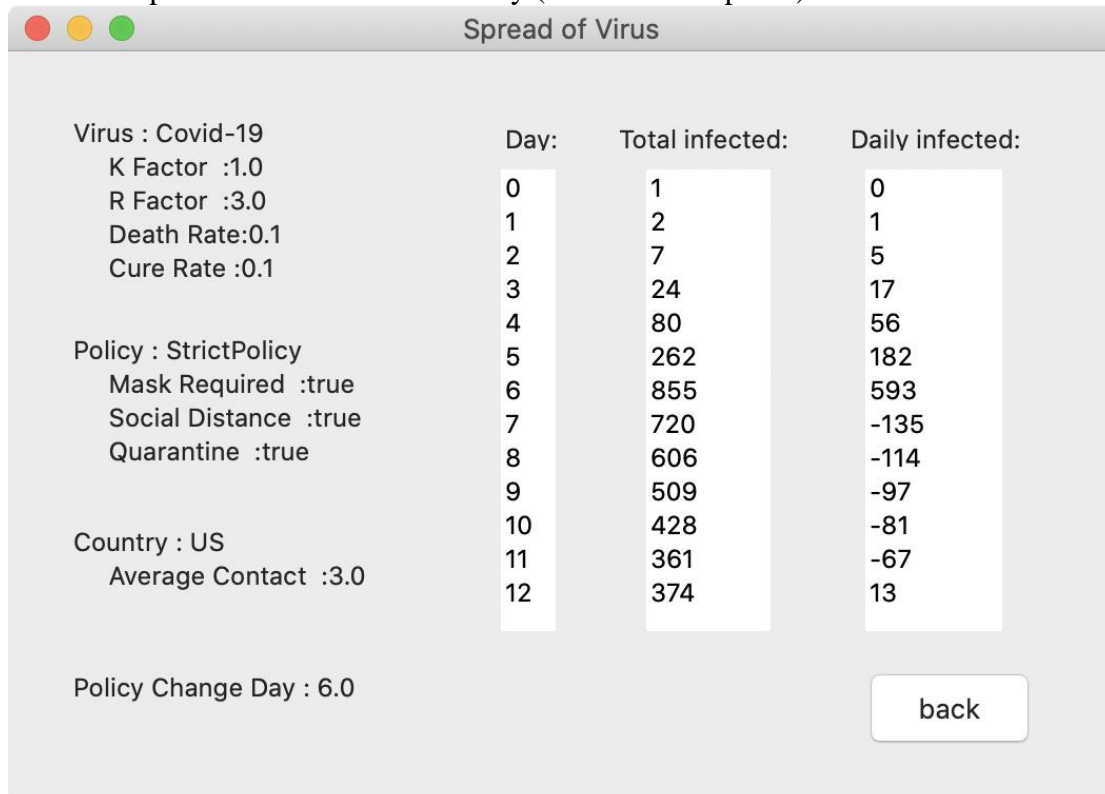
Covid-19 spread in China with single policy (Social Distance):



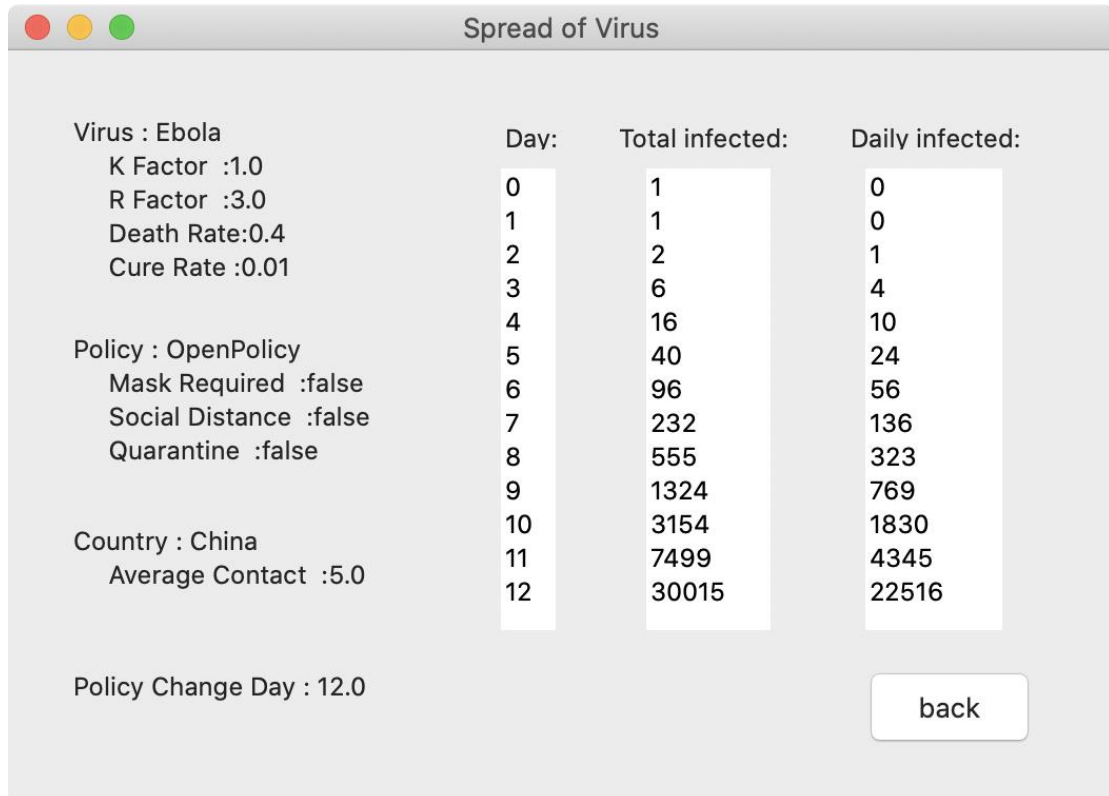
Covid-19 spread in U.S with OpenPoilcy (no measures spread):



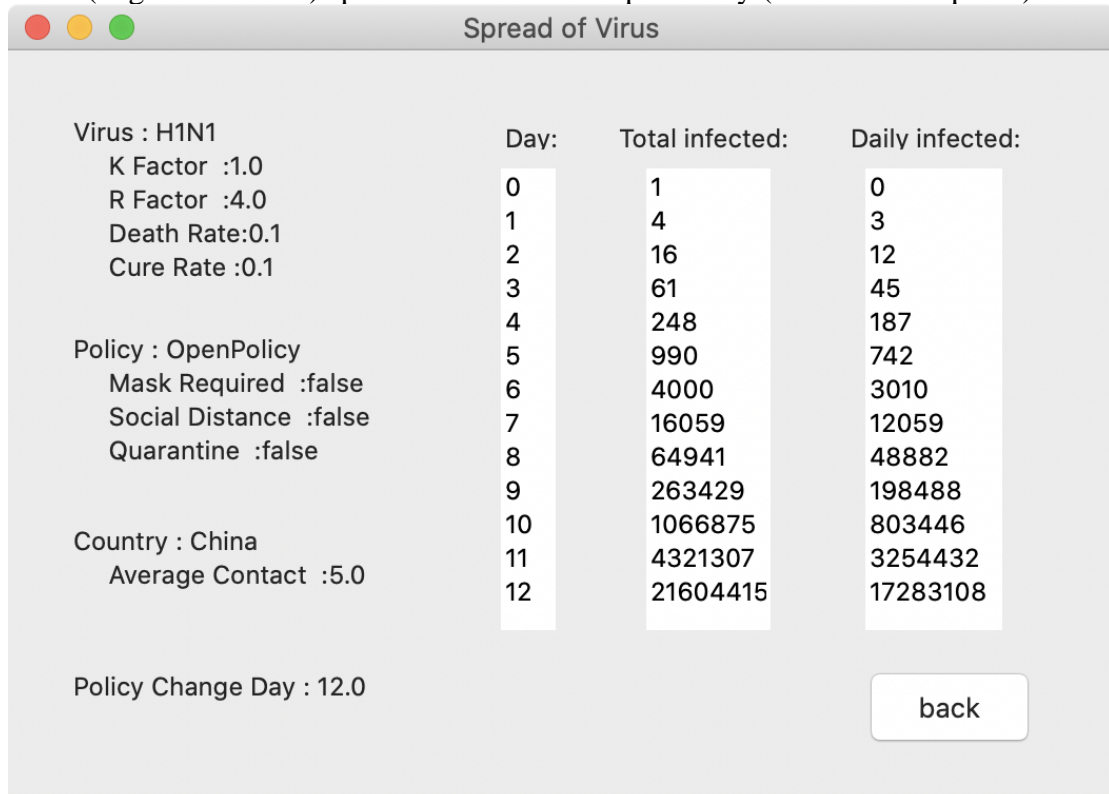
Covid-19 spread in US with StrictPoilcy (no measures spread):



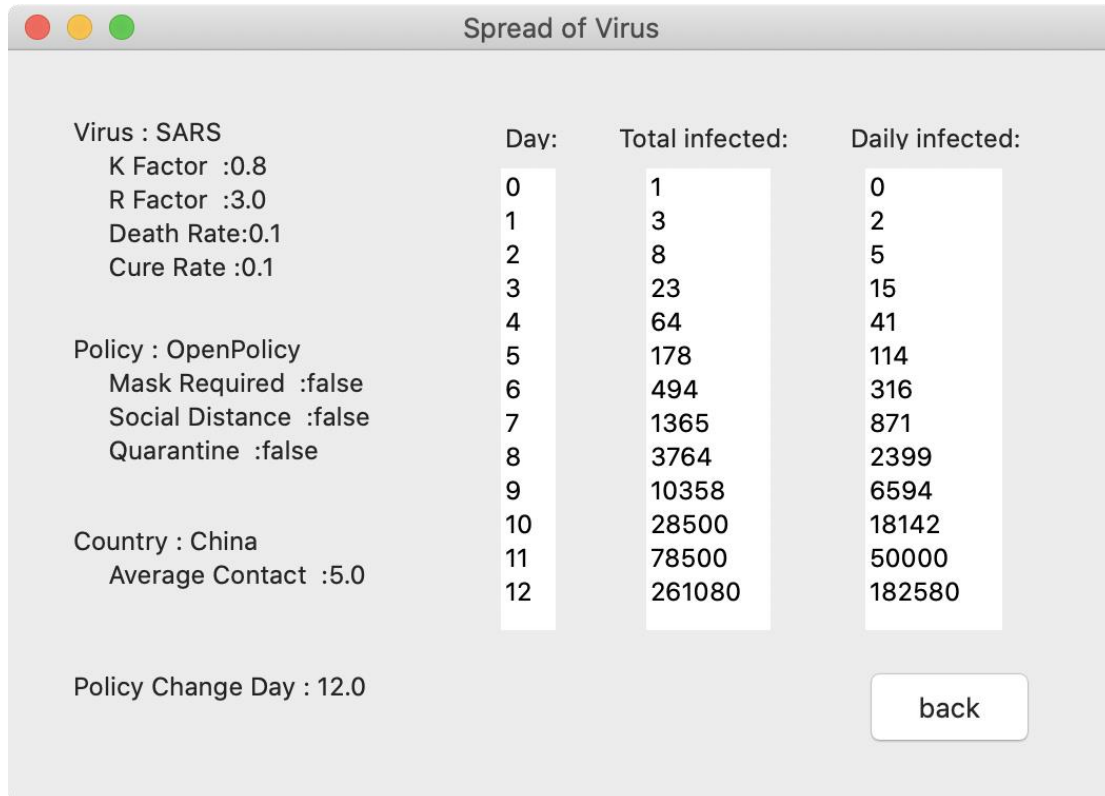
Ebola (High Death Rate) spread in China with OpenPoilcy (no measures spread):



H1N1(High Value of R) spread in China with OpenPoilcy (no measures spread):



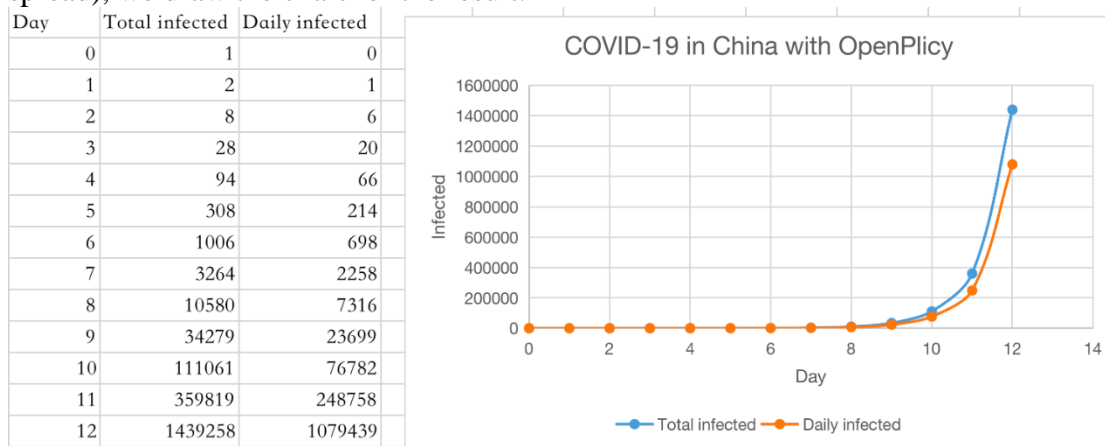
SARS (Low Value of K) spread in China with OpenPoilcy (no measures spread):



6. Mathematical analysis/evidence

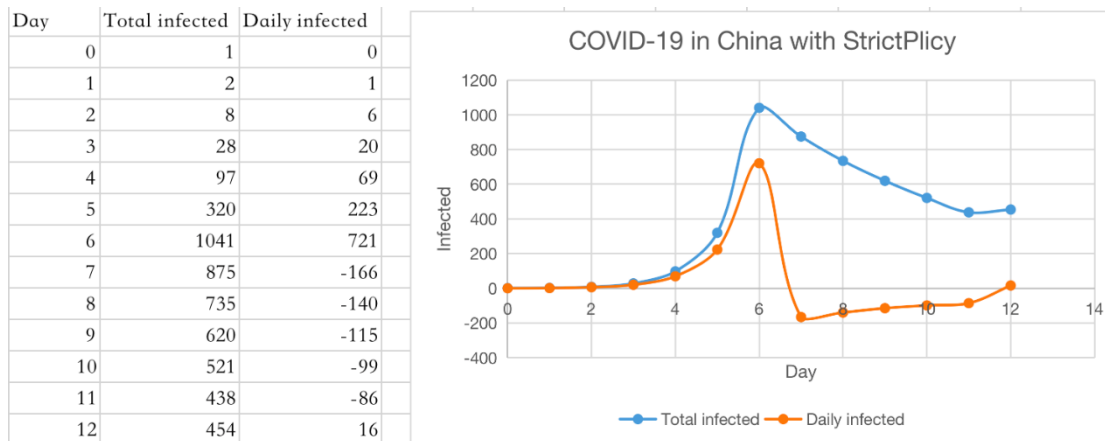
6.1 Same country with Different Policy

First, we simulate the Covid-19 spread in China with OpenPolicy (no measures spread), we draw the chart for the result.



We can see that We can clearly see that without any impact, the virus will spread exponentially. For the first few days, the increase number of infected people was not noticeable. But as the number of simulated days increased, the number of people infected quickly grew and spiraled out of control.

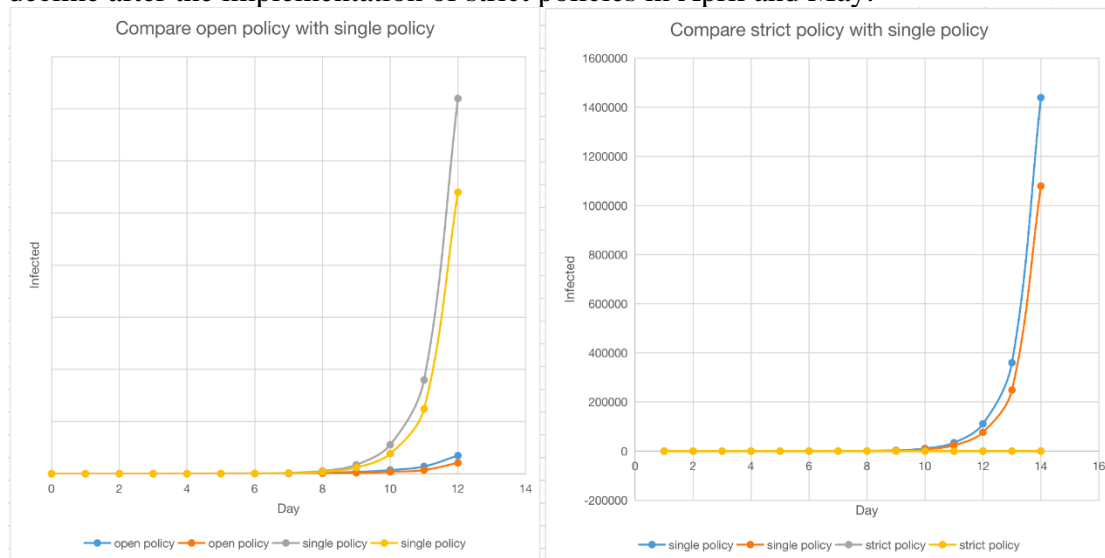
To compare the impact of policies on the number of people infected. We did a second simulation, COVID-19 spread in China with Strict Policy. In this policy, facemask, quarantine and social distance are required. The chart as follow:



We assume that the strict policy is implemented on the sixth day of the virus transmission, as we can clearly see in this diagram, when the policy is implemented. The number of people infected each day is down significantly. The number of new infections per day can even be negative as previously infected people are cured or die.

This shows that a strict policy has a big impact on the spread of the virus. We then tested simulations of virus transmission under a single policy, each of which had significant implications for the duration of the virus transmission.

Also, this figure could explain why in reality China (Wuhan) saw a sharp decline after the implementation of strict policies in April and May.



In addition, the conclusion of a single policy also tells us that a single policy is not enough. Although it can slow down the spread to a certain extent, the number of infected people will continue to increase, and only strict policies can completely curb the spread of the virus.

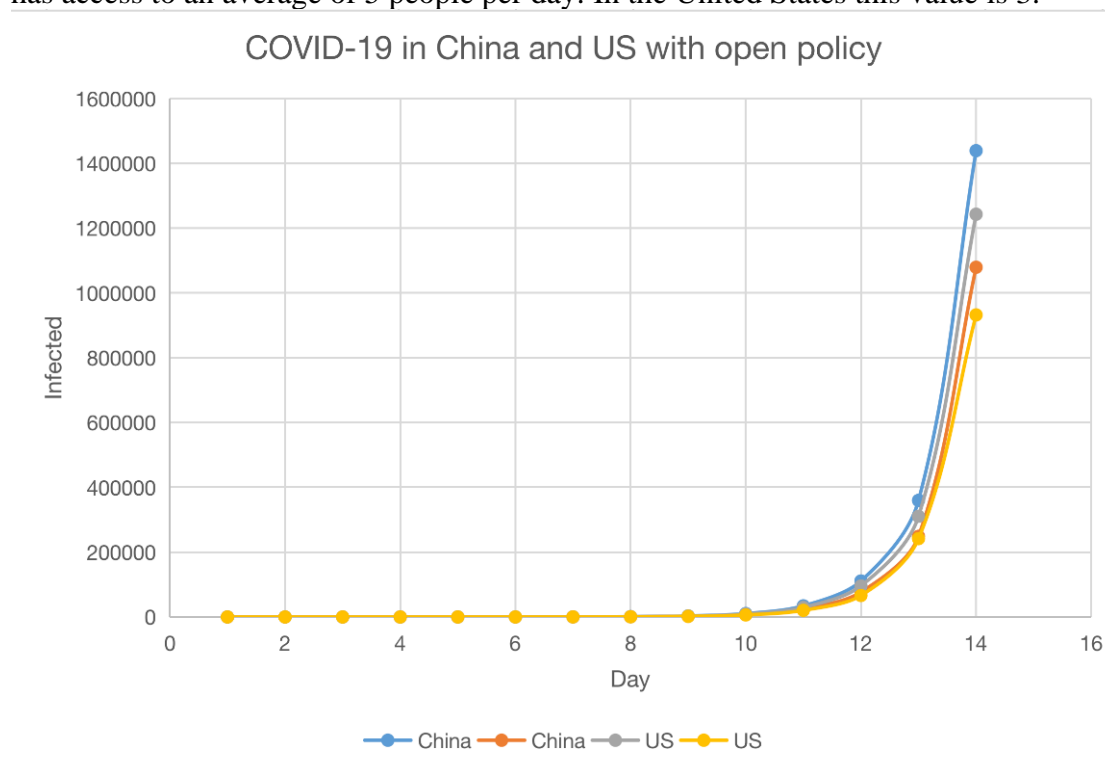
6.2 Different country with Same Policy

To simulate the spread of the virus in different countries, we ran a simulation of the same policy for the United States and China.

Different countries have different population densities, which can lead to significant differences in the speed and extent of virus transmission. In our simulation, we use different Average contact to represent different countries.

In China, we set the value of average contact to 5, which means that each person

has access to an average of 5 people per day. In the United States this value is 3.



In this result, we find that the number of infected people in the US is slightly less than that in China under the condition of OpenPolicy. But overall, the spread of the virus in China and the United States is similar. This phenomenon is because there is another factor affecting the transmission-- R .

According to the definition of R , each virus has only one Value of R_0 , which means that the transmission intensity of the virus is similar without interference. Therefore, strict policies are necessary regardless of whether the country density is different. (See reference 1)

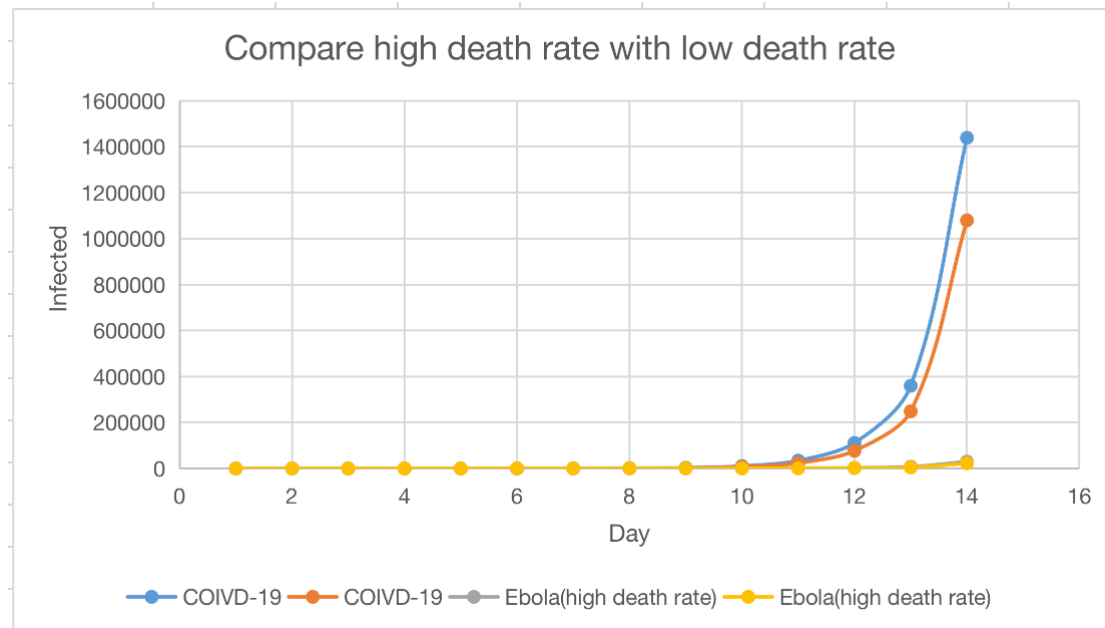
6.3 Different virus with Same Country and Policy

In order to compare the transmission capacity of different viruses, we simulated the transmission results of several different virus with different cure rate, death rate K , and R value in the same environment.

First, we compared two different virus, covid-19 and Ebola.

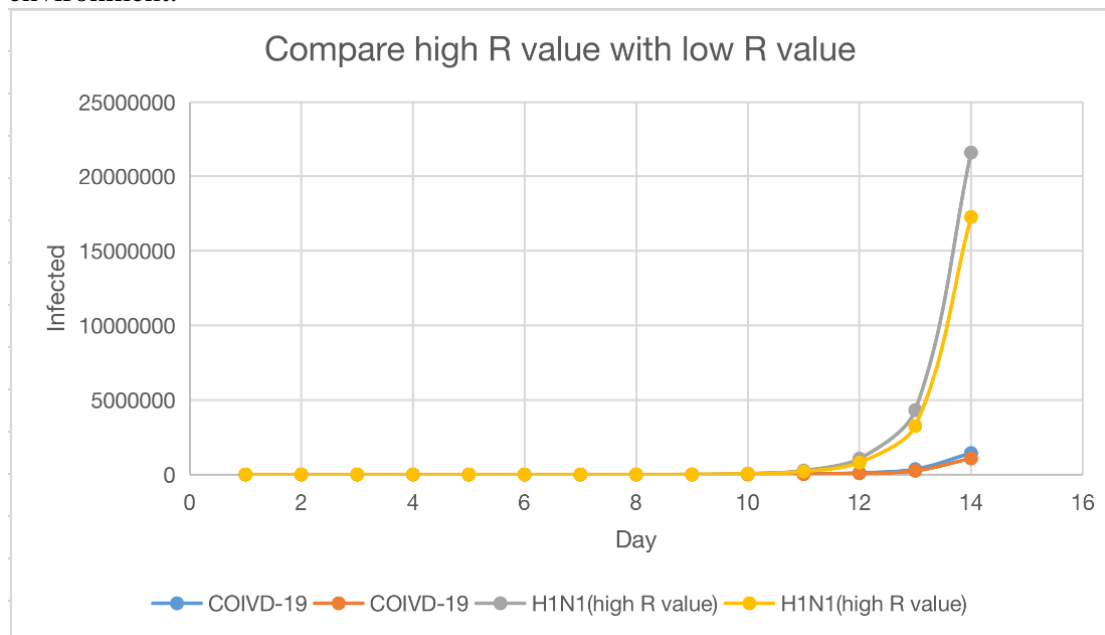
According to this chart, we can find that the virus with high death rate will be less spread ability.

This is because the infected person dies and loses the ability to continue to infect other people. In other words, the number of infected people will be reduced (Compare to the Low Death Rate Virus).



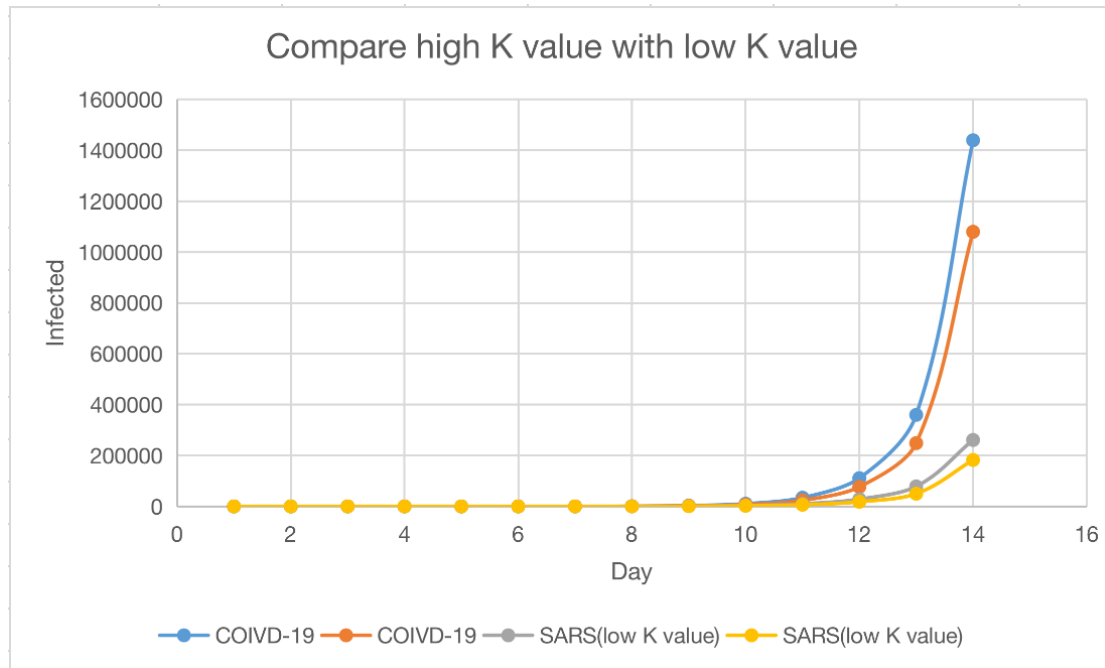
Second, we compared the effects of viruses with different R values on transmission. The higher the R value, the more transmissible the virus has, so as the comparison in the chart shows, H1N1 has a stronger ability of spread than Covid-19.

H1N1 has spread more widely and infected more people in the same time and environment.



At last, we simulated the spread of viruses with different K values. The K value represents the probability that a person can infect others after being infected.

As shown in the graph, the lower the K value, the fewer people the virus could infect at the same time.



7. Conclusion

In the algorithm perspective, using a limited area to simulate the random walk of a limited group of people to simulate the spread of the virus is not a good choice. Because we can define the average contact person from the calculate of R value. Therefore, we use N-ary tree to make the simulation the spread of virus more credible and more reliable.

In the country perspective, no matter what type of the virus (different R/K value, different death/cure rate), appropriate policy is essential for every country in order to restrict or control the spread of virus.

The different effects of different policy as follows:

physical quarantine > wearing masks > social distance. Obviously, multi-policy is more effective than carry out only single policy.

After a period of simulation of virus transmission without policy intervention, although policy intervention is also effective in curbing the development of the virus, it is still far less effective than the intervention in the first place.

So early detection and policy intervention would be the most effective way to stop the spread of the virus.

In the virus perspective, a virus which has a high death rate or a low K value is not good for the spread of itself. Therefore, a perfect virus should have these features: High R value, High K value, Low death rate, Low cure rate.

8. Reference

What is the R number, and why is it relevant to coronavirus? - retrieved from:
<https://www.sciencefocus.com/the-human-body/what-is-the-r-number-and-why-is-it-relevant-to-coronavirus/>

The K factor: Nevermind R, here's the number we need to understand -retrieved from:
<https://www.sciencefocus.com/news/the-k-factor-nevermind-r-heres-the-number-we-need-to-understand/>

COVID-19 CORONAVIRUS PANDEMIC -retrieved from:
<https://www.worldometers.info/coronavirus/>

Coronavirus disease 2019 – retrieved from:
https://en.wikipedia.org/wiki/Coronavirus_disease_2019