Department of Informatics
Technical University of Munich

Connected Mobility Basics

Assignment 1 Report

Group 8
Yigit Kemal Erinc
Gulbike Imge Koksal
Berkay Özerbay

Abstract

Human behavior has always been a popular research area due to its complexity and being composed of many interesting features and characteristics, which are examined by different branches of sciences, ranging from psychology to economy. One of the subtopics of human behaviour is human mobility, which we'll focus on in this report. In recent years, there have been different approaches in computer science to design a mobility model that describes, simulates and analyzes the characteristics of human mobility. specially, there has been a constant effort to design a mobility model that best describes and analyzes all the characteristics of the human mobility.In this project, we implemented a mobility model for for the Department of Mathematics and Informatics building, at Garching Campus and simulated the daily human activity in some aspects with ONE simulator which was developed at Aalto University and is now maintained and extended in cooperation between Aalto University (Comnet) and Technische Universität München (Connected Mobility). [1]

1. Introduction

The aim of this project is to simulate a day of students in the faculty building of mathematics and computer science in Garching and build a corresponding mobility model in ONE Simulator. The model focuses on realistic movements of students inside the faculty building.

Different movement patterns (e.g going to lecture, going to cafeteria) are analyzed and the simulated students show transitions between these patterns in a realistic manner. Meanwhile, the transition of the virus between students and the transition from healthy to infected states of students during their movements in different mask settings (e.g no Mask, medical, FFP-2) can also be observed. At the end of this report we provide various results and their corresponding explanations.

2. Problem Definition

With this work, our aim is to model the speed of transmission of Coronavirus when the subjects are equipped with different types of masks. Our goals are to model the spatial and temporal aspects of different actors (student, professor) in the Informatics building throughout the day and model the transmission of virus while this movement is taking place.

3. Implementation of the Mobility Model

In our model, we have focused on the spatial and temporal behaviours of the students in the FMI building. As a mobility model, we have used "Protected Polygon" to ensure that the movement of the students was limited inside the building. This model contains the map for FMI building and the calculation of the movement paths.

Spatial Aspects

We have decided to simulate the ground floor of the FMI building and modeled the faculty with the help of the OpenStreetMap. We have also defined some specific areas in the building such as lecture hall, hörsaal, cafeteria, library, and computer hall.

Temporal Aspects

We have come up with a daily lecture plan for the student groups. So, students can go to a lesson according to their schedule, remain in the class during the lecture duration which is 1 hour in our case, and then leave the class when the lesson ends. We have also specified different duraitons for different states we have created, which will be discussed in the following section.

States and Transitions

We have defined six different states which can be listed as follows:
- Lecture state
- Study state
- Cafeteria state
- Free time state
- Arrival state
- Departure state

Lecture state is called whenever the scheduled time comes for that node, which causes the node to transition its state from the previous state. Study, cafeteria and free time states are called based on a specified probability.

Detailed explanations of all the states are as follows:

- **Lecture State:** Each student and professor has a schedule of lectures. When the lecture starts, the node moves to the location that the lecture is being held. Stays at the lecture hall during the lecture hour and when the lecture ends, the node moves either to Cafeteria State or Free Time State based on the given probability.

- **Study State:** In this state students either choose the library or the computer hall to study. When this state is called, the node moves to the chosen study location according to a specified probability. After a certain amount of study time, the node transitions its state to the Free Time State.

- **Cafeteria State:** In this state, students or professors go to the cafeteria in the FMI building. After a certain amount of eating time, the node transitions its state to the Free Time State.

- **Free Time State:** In this state, the node moves to a random location inside the FMI Building to get some free time. After a certain amount of time, the node changes its state to Cafeteria State or Study State or remains in the Free Time State based on a specified probability.

- **Arrival State:** During this state students and professors move inside to the building through the main entrance.

- **Departure State:** During this state students and professors exit the building.

4. Integration with The One Simulator

Creating the Map

We have created the .wkt files via OpenStreetMap and created the map and routes files in the simulator. We have changed the default Polygon class (ProhibitedPolygonRwp) to FMIPolygon, entered the coordinates for the new polygon, mirrored it around the X axis, and set the bounds.
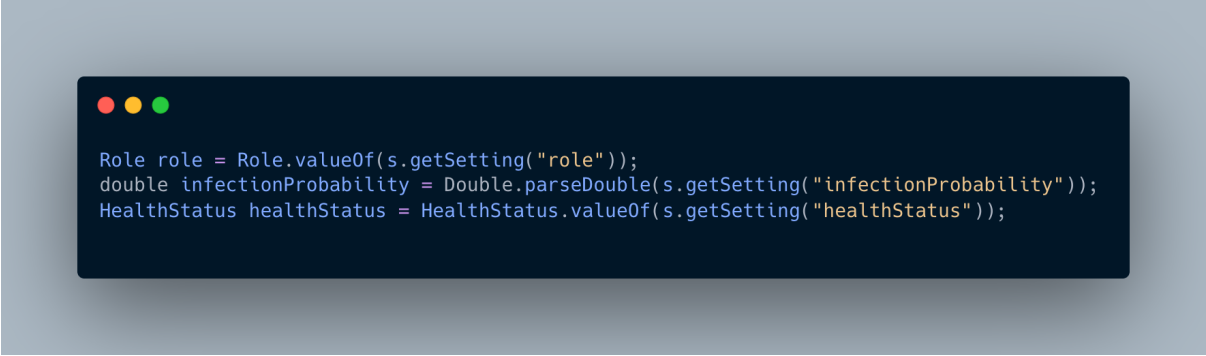
Storing the Infection/Health and Role States of Nodes

To store the additional information about nodes we modified the **DTNHost** class which resembles a node and added the following variables:

```
private HealthStatus healthStatus;
private Role role;
```

These 2 variables are passed to the constructor and set when an object is created. In createHosts method inside SimScenario:

```
Role role = Role.valueOf(s.getSetting("role"));
double infectionProbability = Double.parseDouble(s.getSetting("infectionProbability"));
HealthStatus healthStatus = HealthStatus.valueOf(s.getSetting("healthStatus"));
```

Then the settings are passed to object in creation:

```
DTNHost host = new DTNHost(this.messageListeners, this.movementListeners, gid, interfaces, comBus,
mmProto, mRouterProto, role, healthStatus);
host.setInfectionProbability(infectionProbability);
```

Getting Infected with a Probability

In real life, you do not get infected whenever you get exposed to a virus and the probability is a function of many variables such as distance, the amount of virus you get exposed to (masks affecting this), your immune system and so on.

To mock it to some extent, we have added the **infectionProbability** to the settings as explained in the previous section. The nodes should get infected with a probability after receiving a message. This logic is implemented in the MessageRouter class's **receiveMessage** method.

```java
if (from.isInfected()) {
    this.host.getExposedToVirus();
}
```

getExposedToVirus is the method responsible for changing the state of a node to INFECTED with a probability.

```java
public void getExposedToVirus() {
    if (healthStatus == HealthStatus.INFECTED) return;

    double timePassedUntillLastExposition = SimClock.getTime() - lastExposedTime;
    int THRESHOLD = 250;
    if (timePassedUntillLastExposition < THRESHOLD) {
        return;
    }

    var rand = random.nextDouble(); // Between [0,1]  0.5
    boolean shouldGetInfected = rand <= INFECTION_PROBABILITY;
    if (shouldGetInfected) {
        this.healthStatus = HealthStatus.INFECTED;
    }

    lastExposedTime = SimClock.getTime();
}
```

Changing the Color of Infected Nodes & UI Updates

We wanted to change the colors of infected nodes to red so that the infected nodes can be traced easily while watching the simulation.

This is done in NodeGraphic class.


```
private static Color infectedHostColor = Color.RED;
g2.setColor(node.isInfected() ? infectedHostColor : hostColor);
```

We have also added some modifications to show the current state and health status of the nodes in their names.

Avoiding Multiple Infections In Short Time
While running the simulation we have noticed that, in the case of 2 nodes moving closely for some time, the message gets sent over and over and especially in the lecture state where many nodes are really close to each other, this creates nonsense.

We have shrunk the radius of connections but it did not solve this problem.

To fix this issue, we have added a threshold (in simulation time) that automatically results in a failed infection (exposed to virus but not infected) if the time passed after the last message does not exceed the threshold.

It is in the **getExposedToVirus** method.

```
public void getExposedToVirus() {
        if (healthStatus == HealthStatus.INFECTED) return;

        double timePassedUntilLastExposition = SimClock.getTime() - lastExposedTime;
        int THRESHOLD = 250;
        if (timePassedUntilLastExposition < THRESHOLD) {
            return;
        }

        var rand = random.nextDouble(); // Between [0,1]  0.5
        boolean shouldGetInfected = rand <= INFECTION_PROBABILITY;
        if (shouldGetInfected) {
            this.healthStatus = HealthStatus.INFECTED;
        }

        lastExposedTime = SimClock.getTime();
    }
```

Creating a Custom Event called Exposed To Virus

We have created a custom event to log virus transfers to the panel in **EventLogPanel** class.

```
public void virusTransferred(Message m, DTNHost from, DTNHost to) {
        processEvent(msgTransferStartCheck, "Exposed to virus", from, to, m);
    }
```

In the **receiveMessage** method in **MessageRouter**, we are calling this event after exposing a node to the virus.

```
if (this.host.getHealthStatus() == HealthStatus.HEALTHY) {

    for (MessageListener ml : this.mListeners) {
        ml.virusTransferred(m, from, this.host);
    }
    return DENIED_UNSPECIFIED;
}
```

5. Design Scenarios

We have created 3 different scenarios to model the effect of different types of masks on the spreading of coronavirus, namely: No mask, medical mask and FFP2 mask.

These settings are available on the project root folder named as: nomask.txt, medical_mask.txt and ffp2.txt.

All scenarios share common properties such as: End time (30000), nrofHostGroups (4), worldSize (500, 500) and router (EpidemicRouter).

Group settings and roles are also the same in each settings file. Our groups are:

1) One node to start messaging:
   Number of hosts: 1,
   MovementModel: StationaryMovement

2) Infected Student:
   Number of hosts: 1
   Role: STUDENT
   MovementModel: FMIPolygon
   healthStatus: INFECTED

3) Healthy Students:
   Number of hosts: 10
   Role: STUDENT

MovementModel: FMIPolygon

healthStatus: HEALTHY

4) Healthy Professors

Number of hosts: 3

Role: PROFESSOR

MovementModel: FMIPolygon

healthStatus: HEALTHY

healthStatus is a custom setting that we defined, which gets assigned to each Node/DTNHost's healthy status at the start of simulation. All our scenarios start with a single infected node and a total of 13 healthy nodes, 3 being professors and the other 10, students.

We have 3 events:

1) - From: 1

- To: 0 (0 is the fake destination node for message transmission)

- Class: SingleMessageGenerator

2) - From: 2,2

- To: 0

- Class: SingleMessageGenerator

3) - From: 3,3

- To: 0

- Class: SingleMessageGenerator

We have another custom setting called **infectionProbability**, in our simulation, the virus transmission takes place probabilistically as in real life and this parameter is the probability of an infection after a virus transmission event (after a node gets exposed to virus).
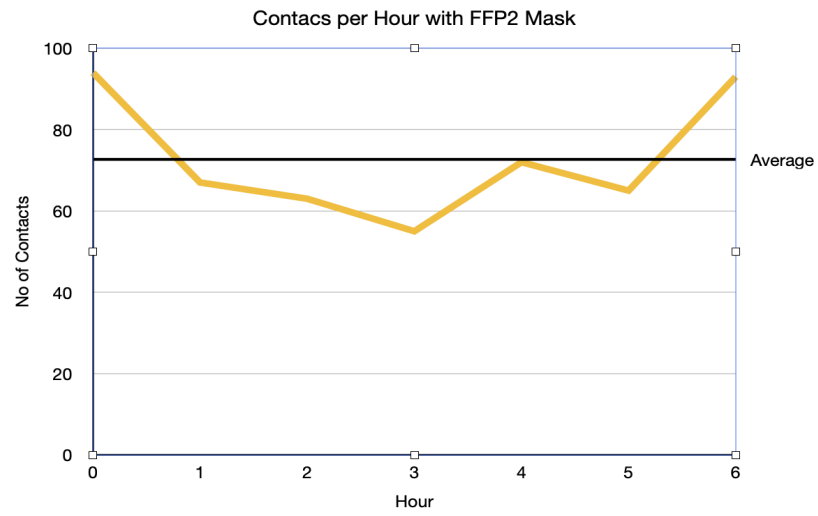
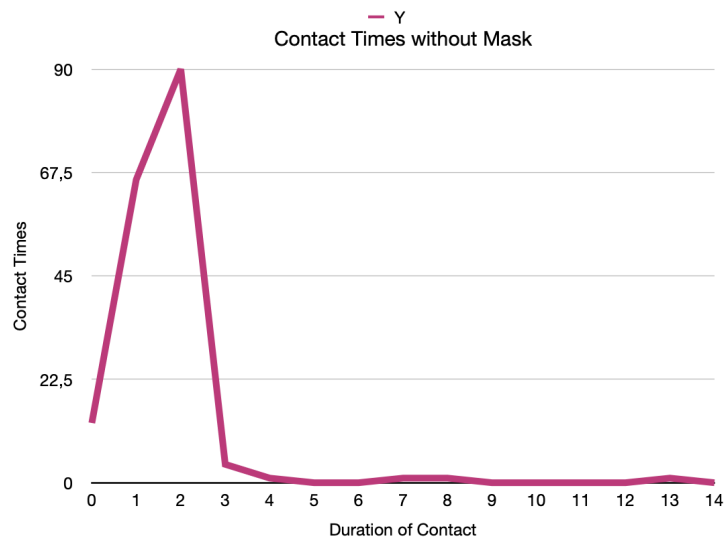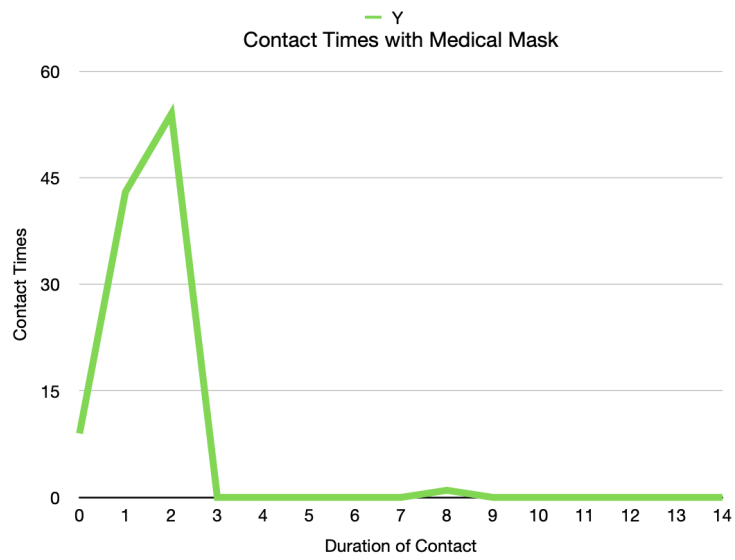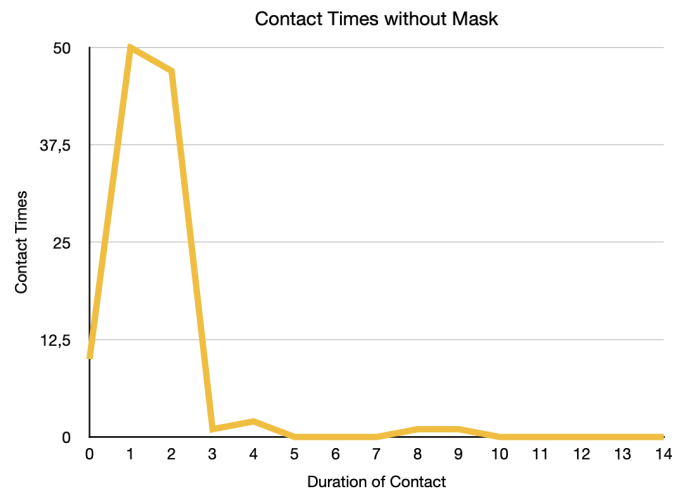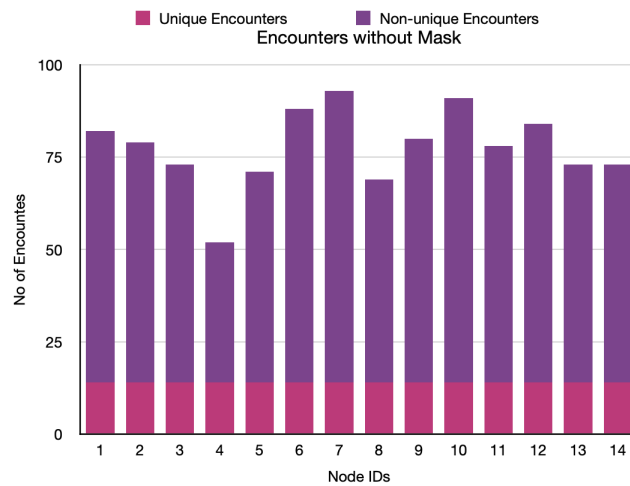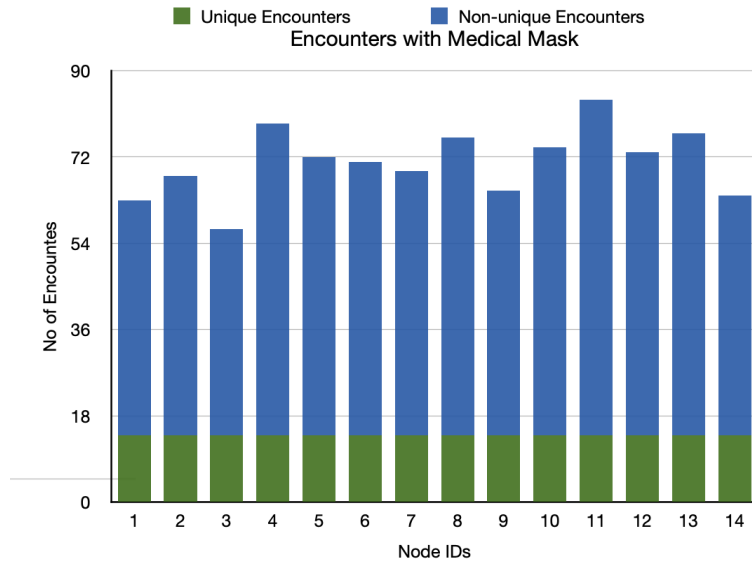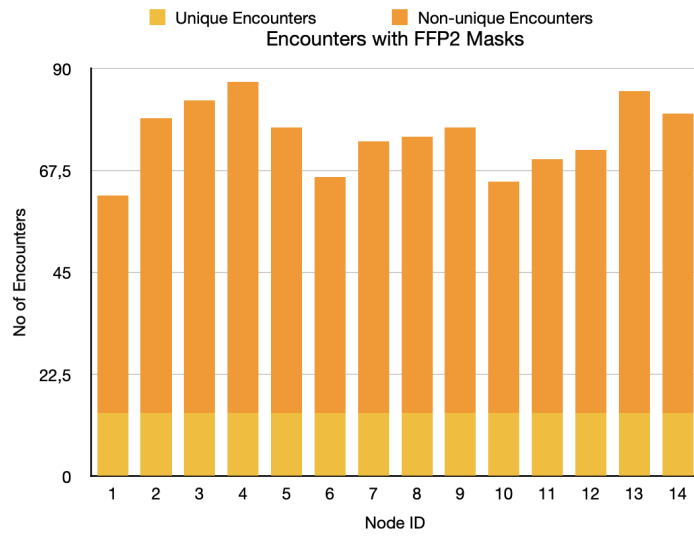| Mask Type	No Masks	Medical Mask	FFP2
infectionProbability	0.3	0.1	0.001

FFP2 - infectionProbability: 0.001

Medical Mask - infectionProbability: 0.1

No Masks - infectionProbability: 0.3

6. Simulation Results

Contacs per Hour with FFP2 Mask

Contacs per Hour with Medical Mask

Contacs per Hour without Mask

Contact Times without Mask

Contact Times with Medical Mask

Contact Times without Mask

Encounters with FFP2 Masks

- Unique Encounters
- Non-unique Encounters

Encounters with Medical Mask

- Unique Encounters
- Non-unique Encounters

Encounters without Mask

- Unique Encounters
- Non-unique Encounters

Contacts Per Hour

The duration of the simulation (in seconds) is divided into hours, and for each hour of the simulation, the total number of connection establishments is recorded. This information showed us the periods of time that the users are more active and moving around in large areas. Although there can be fluctuations, we can see that making contacts is much more possible at the beginning of the simulation than the other parts of the day.

Encounters vs Unique Encounters

These graphics show for each node in the simulation the total number of encounters with other nodes, and the number of unique encounters with other nodes. As you notice, the number of unique encounters is equal to (the number of nodes -1) , so every node has been in contact minimum once with all of the remaining nodes.
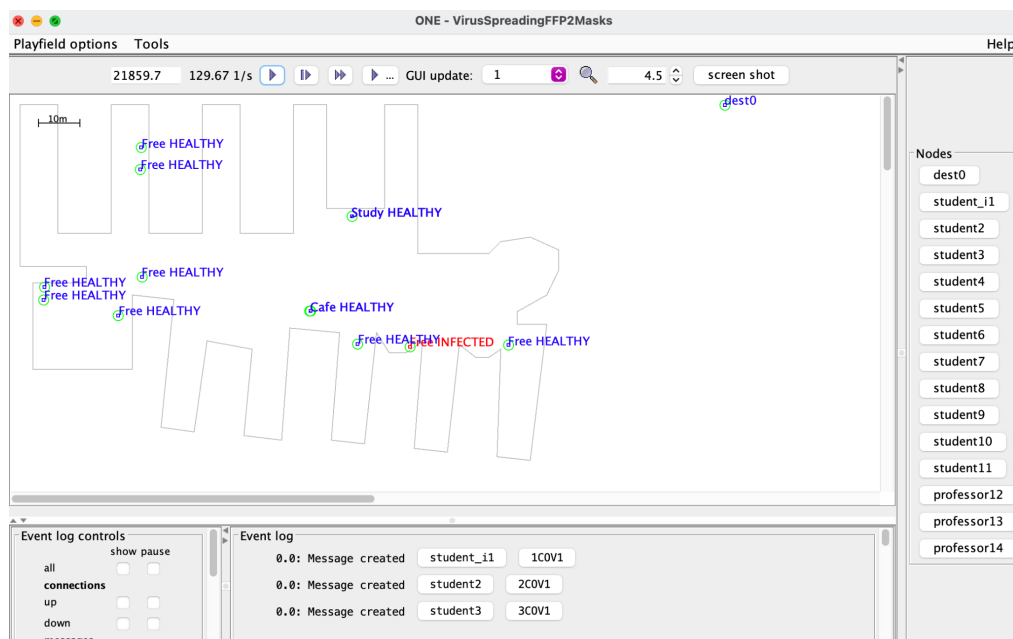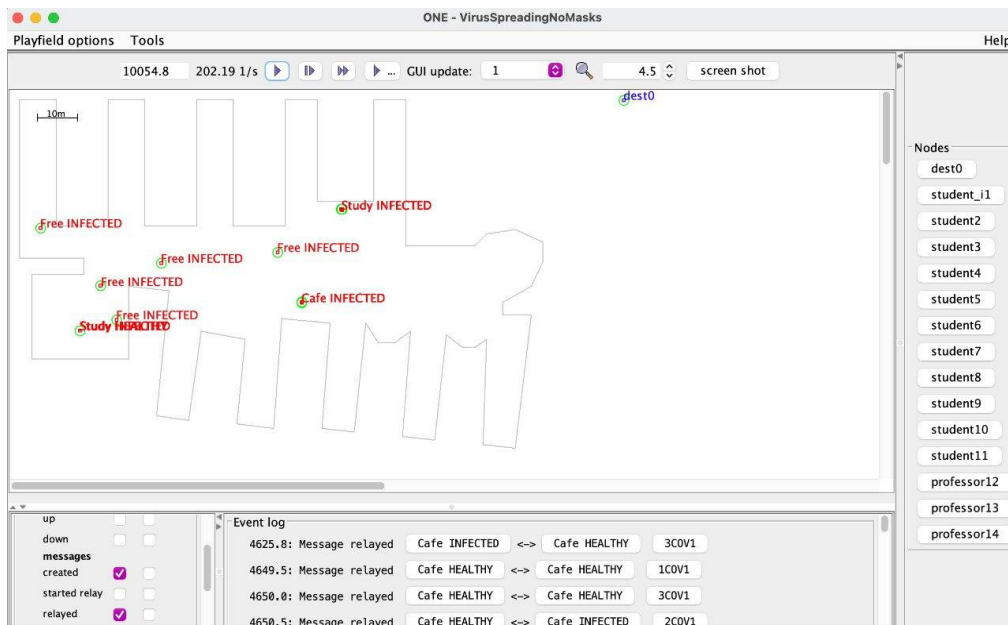
Contact Times

This distribution graphic represents the duration of any connection between any two different nodes. Statistically this graphic shows the usual duration for a connection between two nodes.

Transmission of Virus
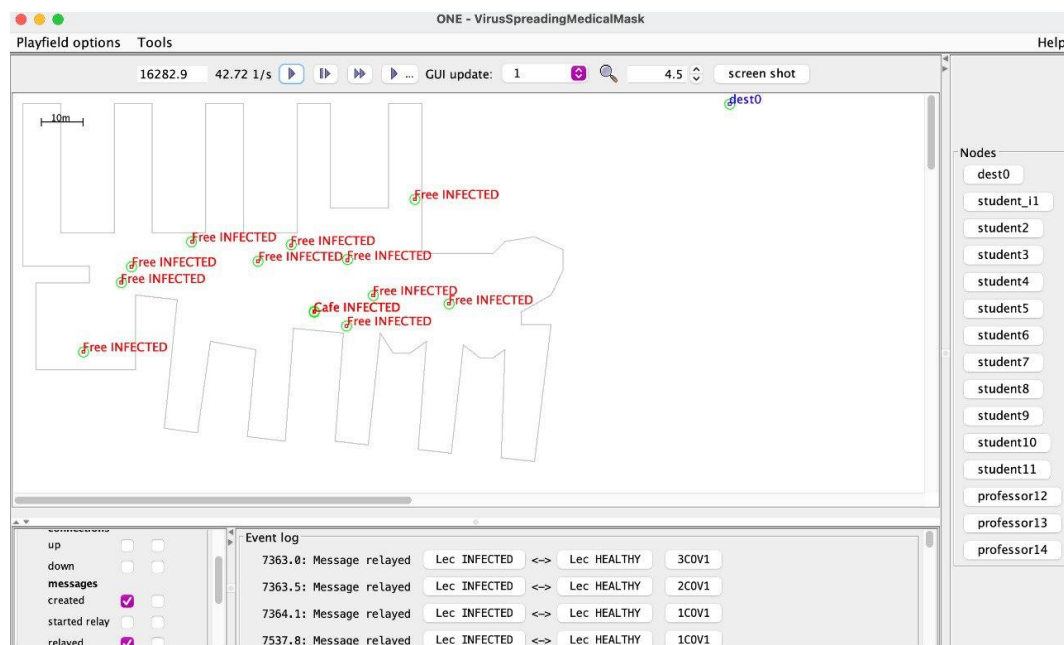
Screenshot of Simulation with medical mask

Screenshot of Simulation without mask

Screenshot of Simulation with FFP2 mask

As you can see on the screenshots above, the simulation time, when all nodes are infected, is around 16200 if the people wear medical masks whereas it is around 10000 if nobody wears a mask. Since there is 0.001 probability of infection if people wear FFP2 masks, nobody got infected at the end of the simulation, that can be seen above. To sum up, 3 type of simulations with different mask configurations showed us the supremacy of FFP2 masks over medical masks and medical masks couldn't make a huge difference compared to having no mask.

7. Conclusion

In our scenario, the aim was to observe and compare the virus spreading among the people inside the Faculty of Mathematics and Informatics based on their mask type. We have utilized the Protected Polygon model to limit human movement inside the building. Our model includes spatial and temporal aspects of human mobility. Each student and professor have a lecture schedule and acts according to that. We have made use of different states that serves as different actions like studying at the library or eating at the cafeteria. We have observed eight hours of movement in a day for three different scenarios (no mask, medical mask, FFP2 mask) and made inferences according to that.

To make our model more realistic, these following can be done as a future work:
- The status of being vaccinated can be taken into account
- Changing probabilities according to distance between nodes can be specified
- Social distance in classrooms can be implemented
- Nodes can be infected according to different factors like age and immunity
- Different states and areas can be added

8. References

[1] Ari Keränen, Jörg Ott, Teemu Kärkkäinen.The ONE Simulator for DTN Protocol Evaluation, Helsinki University of Technology (TKK) (http://akeranen.github.io/the-one/)