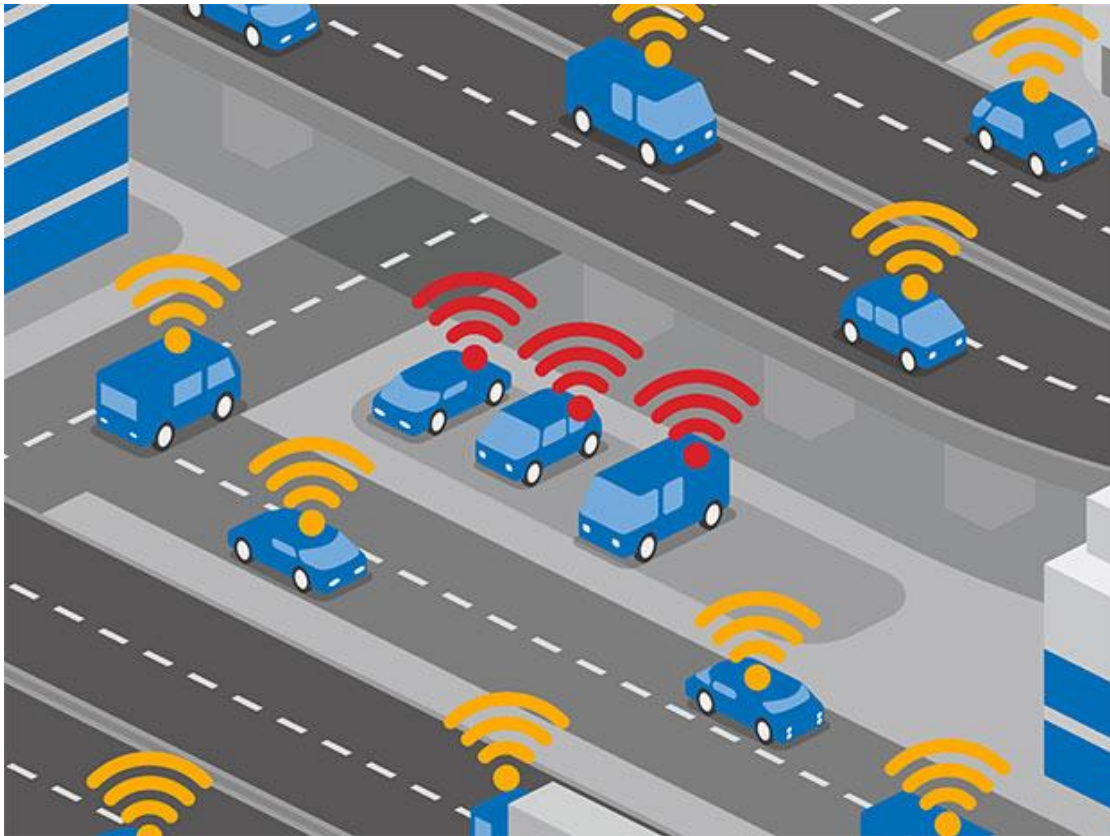


Report Multi-Agent Systems Project

VANET Cybersecurity simulation



Supervised by :

Dr. Zargayouna Mahdi

Elaborated by :

Souhir Arous

Romuald Motcheho Kamguia

Ehab El-kady

Ayoub Bouallagui

Master 2 SIA - 2021 / 2022

Table of content :

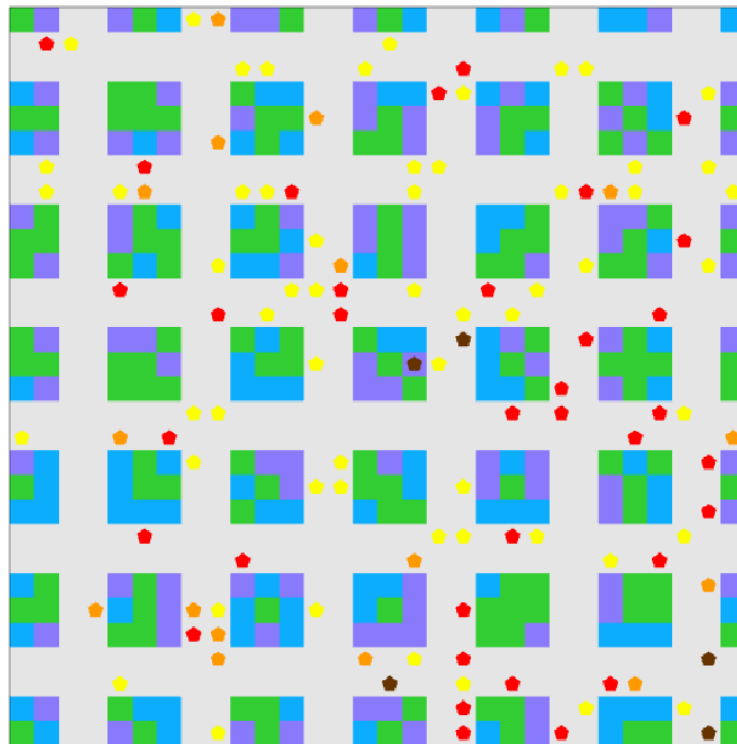
Problem Description	3
System Parameters	3
Movement Scenarios	5
Vehicles Behaviour	7
Results	8
Difficulties Encountered	18
User Manual	19
Conclusion	21

1. Problem Description





The system simulates a cybersecurity attack over vehicles in an environment that has various attractors. The vehicles are divided into four types which are infected, not infected, repaired and broken. The infected vehicle can affect the not infected ones and turn them into infected vehicles with a certain probability. Also, the infected vehicles can be repaired or broken down completely with a certain probability for each. In the project, we simulated the movement of the vehicles in the environment having in consideration those probabilities of change for each vehicle.

1. System Parameters



Here is the “neighborhood” we designed with the vehicles represented by circles and the various spaces represented by rectangles.



The vehicles are designed into four different classes that all of them inherit from the main class which is named “Agent”. The classes are:

-  “Non Infected Vehicle”
 - Colored in Yellow
 - This vehicle is the one that is not infected.
 - On each movement of the vehicle, there is a probability of having an infected one which will affect it to be infected with a probability of 10%.
-  “Infected Vehicle”
 - Colored in Red
 - This vehicle is the one that is infected.
 - On each movement, there can be a noninfected vehicle which will have the effect on with the up mentioned probability and turn it into an infected one.
 - Also, there is another two probabilities on moving which are the probability of getting repaired which is 10% and another probability of being broken down by 5%
-  “Repaired Vehicle”
 - Colored in Orange
 - This vehicle is the one that got repaired from infection
 - On each movement of that vehicle, there is a probability of getting infected again as it became a noninfected vehicle but this probability is less than the one that didn’t get infected at all as there can be an antivirus or a batch installed to protect it from further infection. That probability is 5%
-  “Broken Vehicle”
 - Colored in Brown
 - This vehicle is the one that broke down completely from infection
 - This vehicle has no movement as it is broken down completely which means it does not move

Also, there are other parameters in the system which form the neighborhood where everything take place:

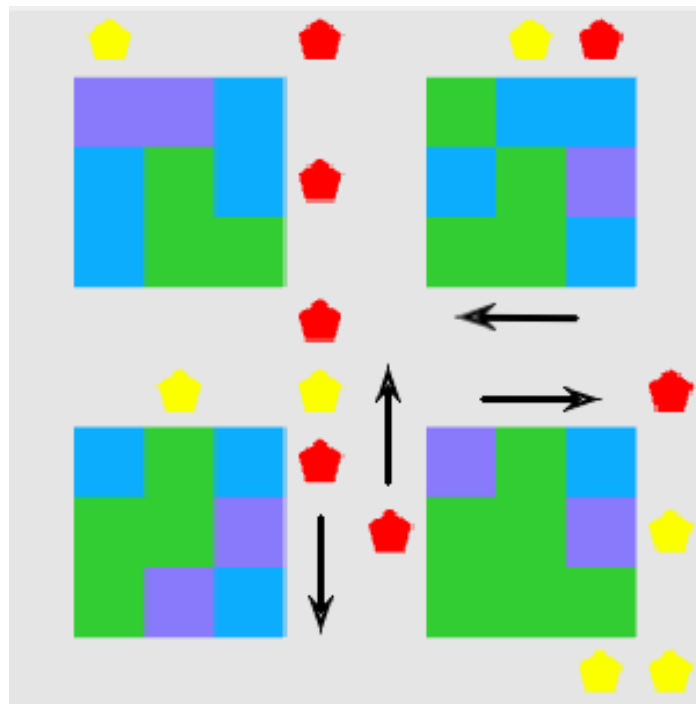
-  “Road”
 - Colored in Grey
 - Where the vehicles move
-  “Public Spaces”
 - Colored in Purple

- Where multiple vehicles can go
- ■ “Private Spaces”
 - Colored in Blue
 - Where only one vehicle can go
- ■ “Gardens”
 - Colored in Green
 - Where no vehicle can go

To be more precise, we didn't implement a class for each type of space, we implemented only one class named 'Element'. The class has an attribute 'type' and this attribute can take a string among "public", "private" and "garden". However, we do have a class 'Road' and all classes inherit from 'Agent'.

2. Movement Scenarios

We implemented the vehicle's circulation as if the roads had directions. In the following figure we can see the direction of each road.



Thus, the movement of a vehicle depends on two conditions :

1. The position :

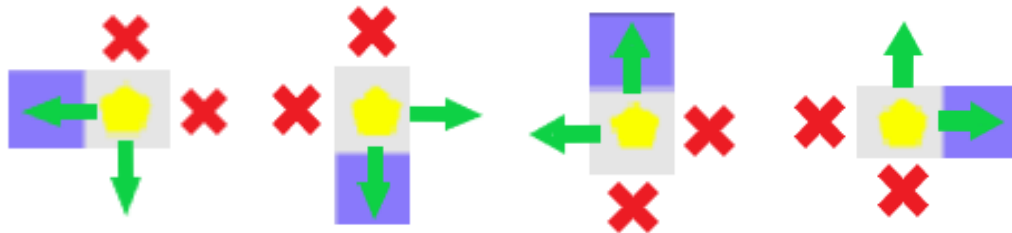
To implement the movement of a vehicle we first need to determine its position. Depending on that position, we will choose the vehicle's new direction. There are ten possible positions categorized in three main positions :

a. Inside a private or a public space



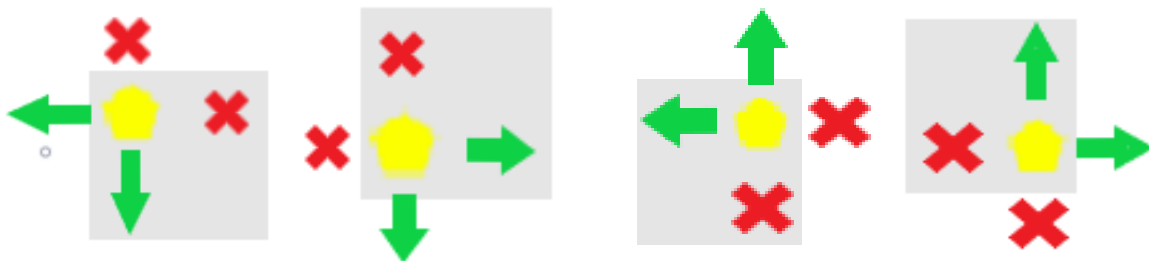
b. Next to an Element

Depending on where the element is placed, we can define the directions the vehicle can take. In the following figure we can see the different directions. Every time we have two possible directions, the choice is made randomly with equal probabilities.



c. Inside what we can call a “roundabout”

No close element but following the same mechanism, we also have two possible directions and the choice is made randomly with equal probabilities.



2. The element

Once we know the position of the vehicle, if we chose a direction that leads to a road there is no problem but if the chosen direction leads to another element, we need to verify its type :

a. “Public” element

valid direction choice. No maximum number or limit or condition on the number of vehicles.

b. “Private” element

Check the number of vehicles in the space because in private spaces we can't put more than one vehicle. If no car then valid choice, else take the second direction.

c. “Garden” element

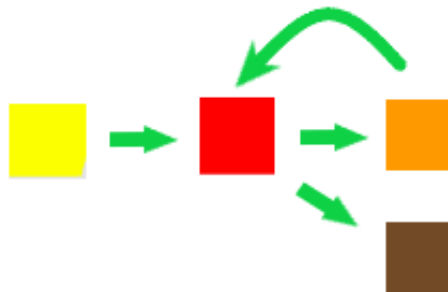
Invalid choice because we can't put any vehicle in garden spaces. So the second direction will be taken.

3. Vehicles Behaviour

The simulation holds all those vehicles and simulates their movement in the environment having all the other elements of the environment mentioned before available there. The simulation is as follows:

- The environment is initiated at first with a random number of infected vehicles and non infected vehicles.
- Each of those cars move in the environment following the mechanism described above.
- In case an infected vehicle hits a non infected one, we get a random decimal and check if that decimal is less than that probability we have set. In that case, we turn the non-infected vehicle into an infected one and we remove the non-infected vehicle from the context.
- On each movement of the infected vehicle, we do the same thing for the probability of getting repaired. we remove the infected vehicle from the context and replace it with a repaired vehicle.

- Also, we check the probability of getting broken down and if the condition applies, we remove the infected vehicle from the context and replace it with a broken vehicle.
- The repaired vehicles start to move in the environment and have the same scenario as the non-infected vehicles.
- The broken vehicles remain motionless for a period equivalent to five movements then disappear and they have no probability of getting repaired again.
- The system keeps going with those conditions and simulation.



4. Results

First, we started with a standard version with standard parameters :

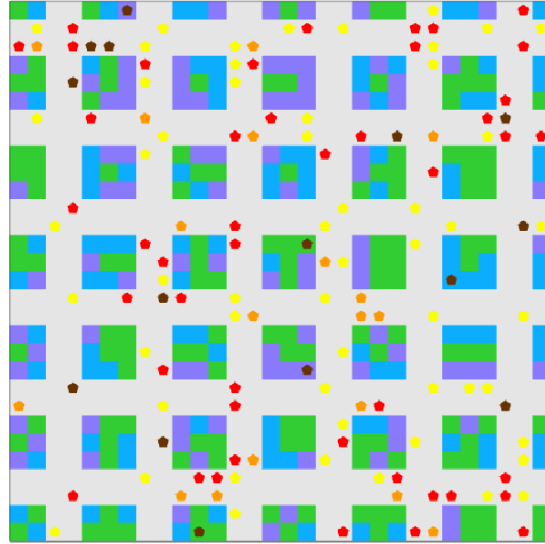
Number of infected and non infected vehicles = Random with equal probabilities

P_{inf} : Probability to get infected for non infected vehicle and repaired vehicle when contact with infected vehicle = 50%

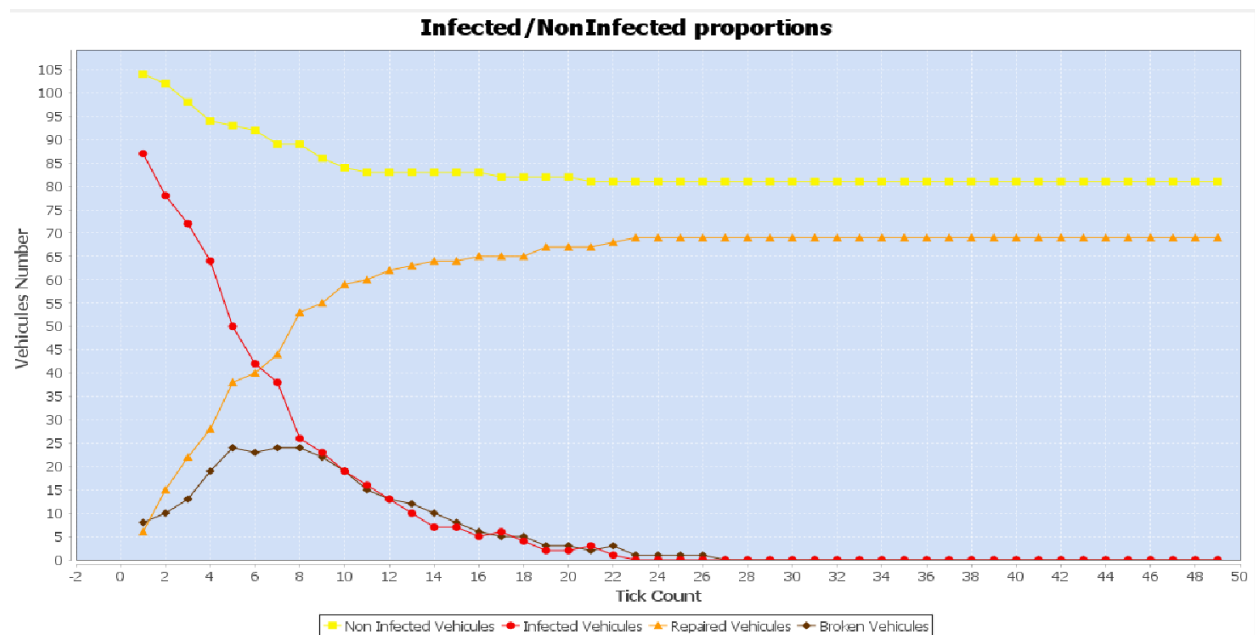
P_{rep} : Probability to get repaired for infected vehicle =10%

P_{break} : Probability to become broken = 10%

Neighborhood structure and conditions of entry to spaces as described above



To see and analyze in real time the evolution of the results we use a chart that describes the number of each type of vehicle. The following chart shows the change in the number of vehicles through the tick counts :



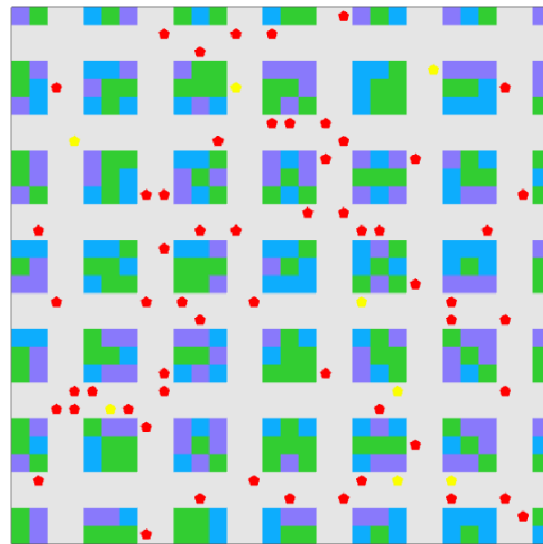
As we can see from the chart, the infected vehicles decrease by time mainly but they can experience a little of increase as vehicles can get infected. While the repaired vehicles increase by time as the vehicles can get repaired which is reasonable with the decrease of the number of infected vehicles. In the same way, the broken vehicle also experiences an increase. Since a broken vehicle is removed from the context, over time, the infected vehicles are finally all deleted, and since a

non infected vehicle or a repaired one can only be infected by contact with an infected vehicle, the numbers of non infected and repaired vehicles become constant.

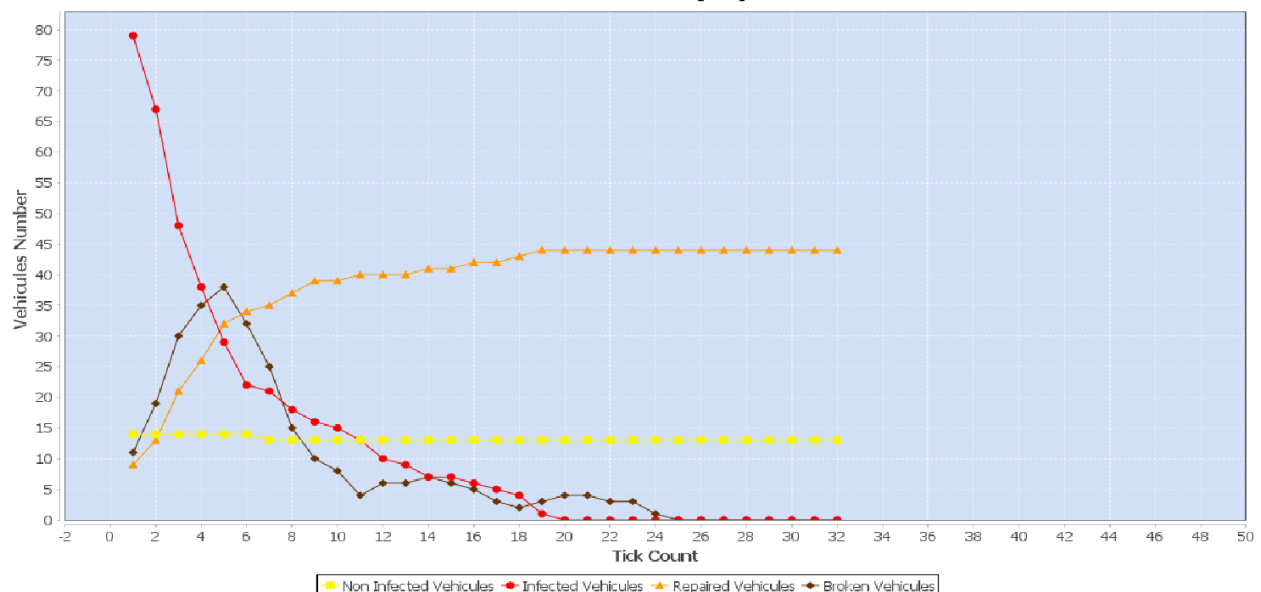
Second, to better analyze the influence of each variable we have made several versions changing each time a chosen variable but always using the first standard version as a base.

1. Proportion of infected/non infected vehicles :

- Increase number of infected vehicles
- Decrease number of non infected vehicles

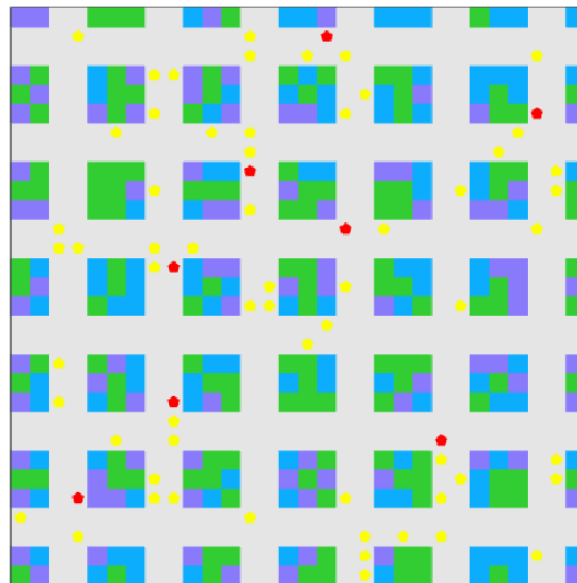


Infected/NonInfected proportions

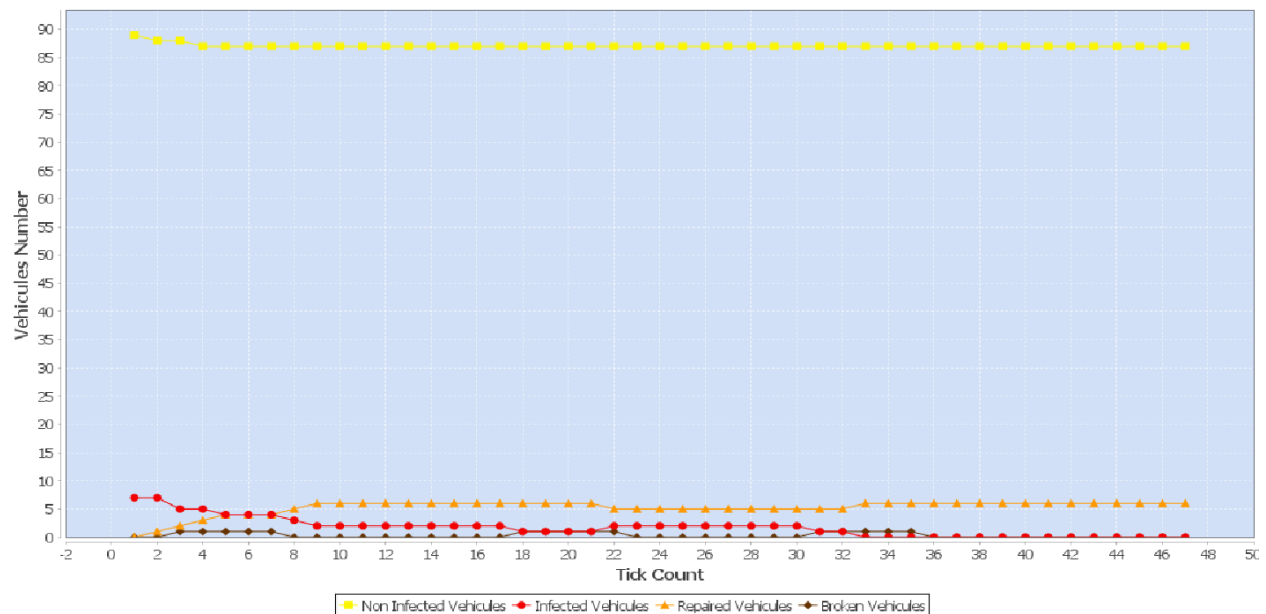


Since the number of non infected vehicles has decreased, the number of infected vehicles that were non infected decreases proportionally. For that reason, the number of infected vehicles becomes zero much faster.

- Decrease number of infected vehicles
- Increase number of non infected vehicles



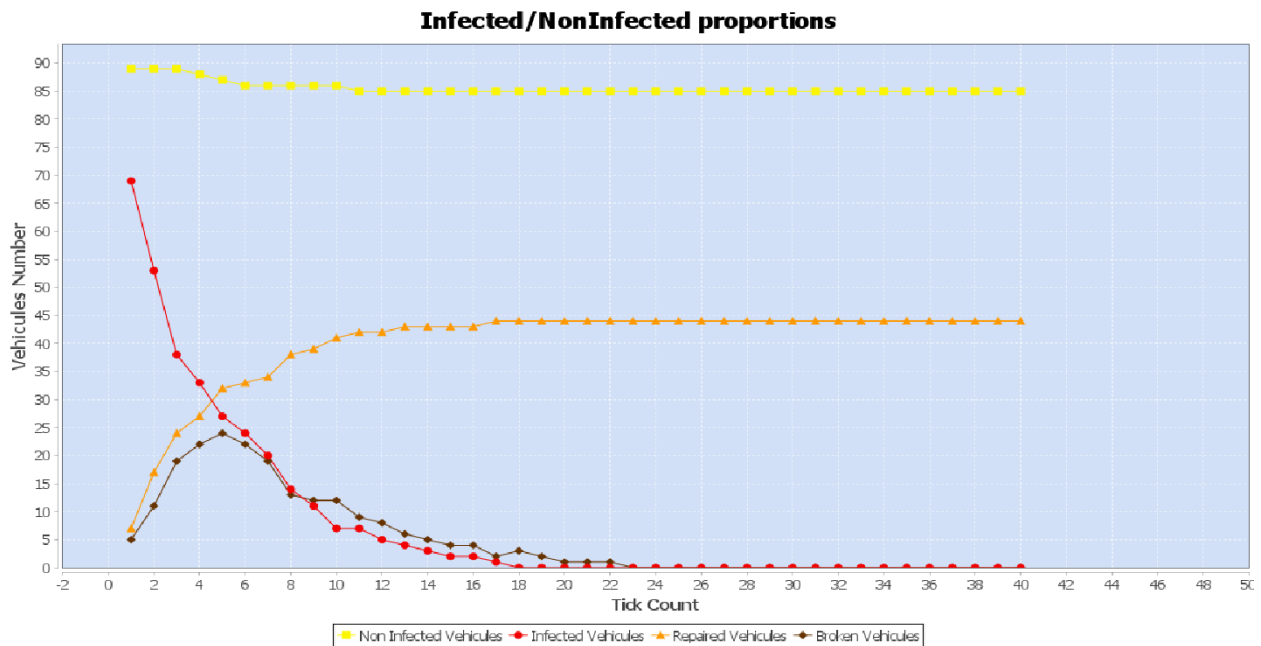
Infected/NonInfected proportions



Since the number of infected vehicles has decreased, the non infected vehicles are less likely to come into contact with an infected vehicle and become infected too. For this reason the number of non infected vehicles remains very high.

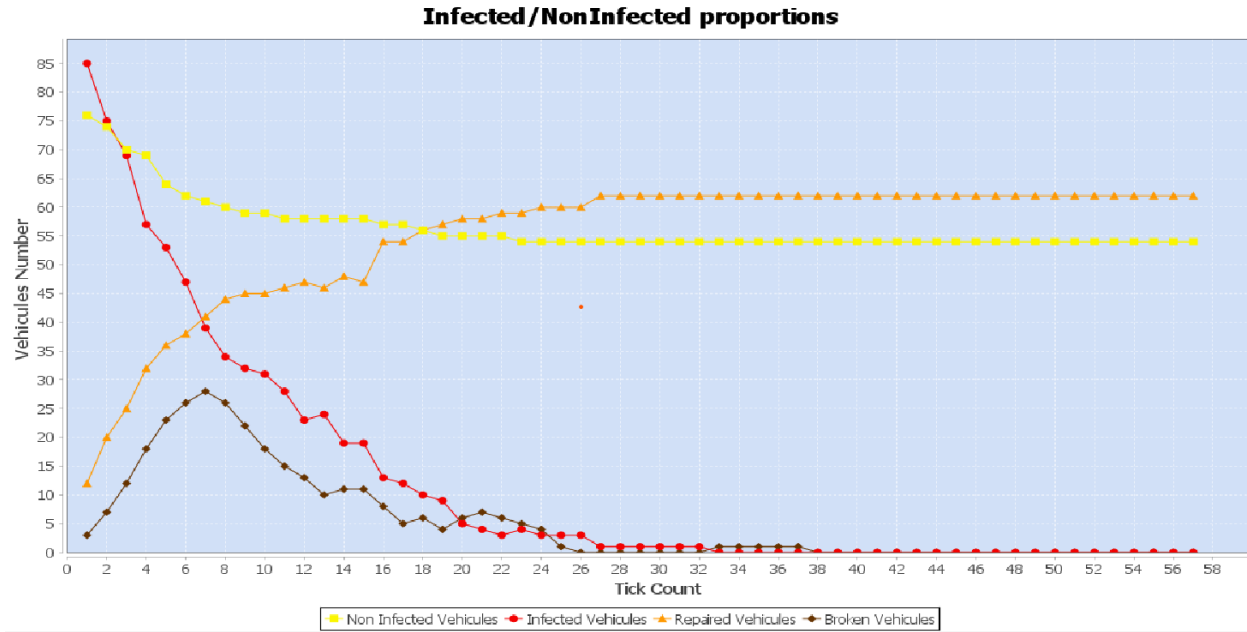
2. P_{inf} :

- Decrease $P_{inf} = 10\%$



Compared to the chart we got with the first version, we can easily see that the number of non infected vehicles decreases much less (85 here compared to 75 in the first version). But as the number of non infected vehicles decreases, the number of infected vehicles decreases proportionally. Therefore, the number of repaired vehicles increases (45 here compared to 70).

- Increase $P_{inf} = 100\%$

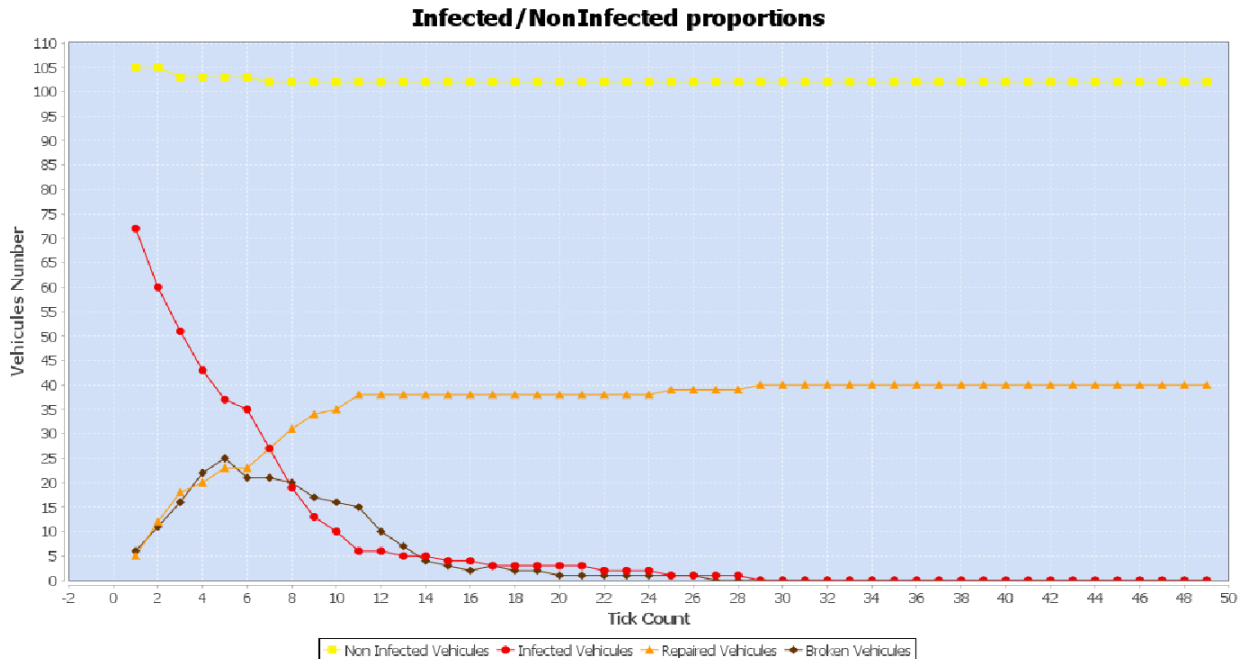


Non infected vehicles are more likely to be infected so their number decreases much faster (53 here compared to 75 in the first version). The number of infected vehicles increases proportionally. For this reason the number of broken vehicles increased (28 here compared to 23 in the first version). The choice between broken and repaired is random, for this reason in this test specifically the number of broken vehicles increased instead of the number of repaired vehicles and that is also why the number of infected vehicles becomes equal to zero approximately at the same time.

- P_{inf} proportional to the number of infected vehicles in the same space

In this case we want P_{inf} (the probability to get infected) to increase proportionally with the number of infected vehicles in the same space. If number of infected vehicles increases, P_{inf} increases.

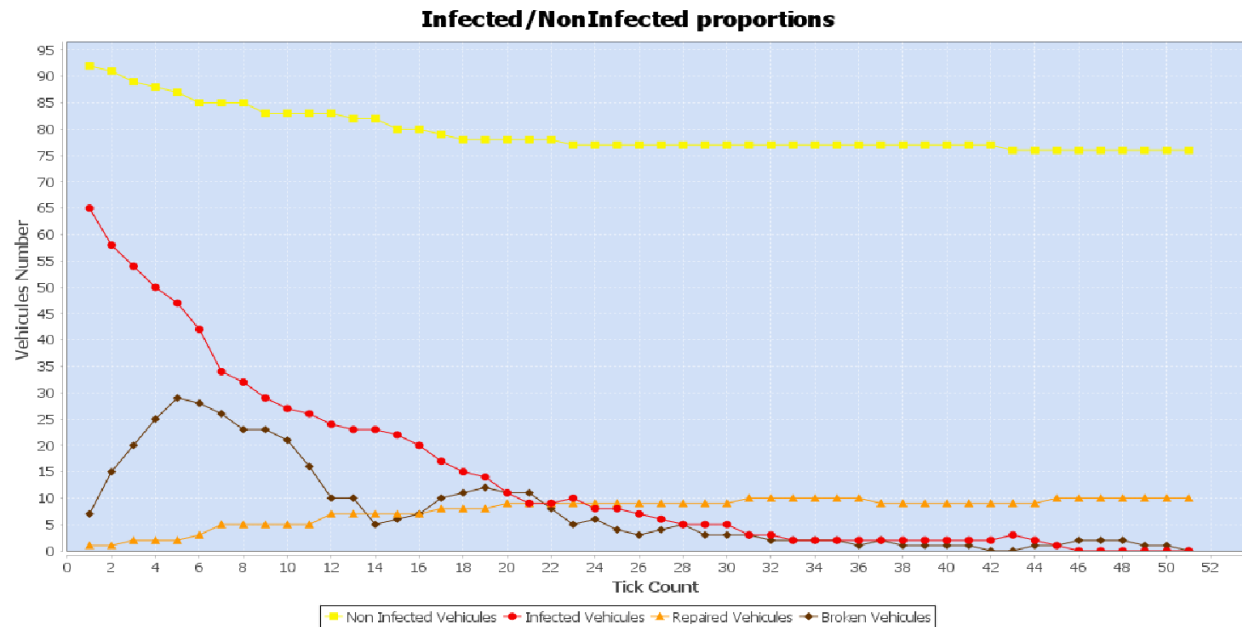
Here is the formula we used : $P_{inf} = 1 - (1 / nbInfectedVehicles)$



Unfortunately, the vehicles are not often numerous in the same space, so the non-infected ones have less chances to be infected. For this reason their number does not decrease much and the number of infected vehicles quickly reaches the value 0. For pedagogical and academic reasons, we choose to determine ourselves the probability of infection

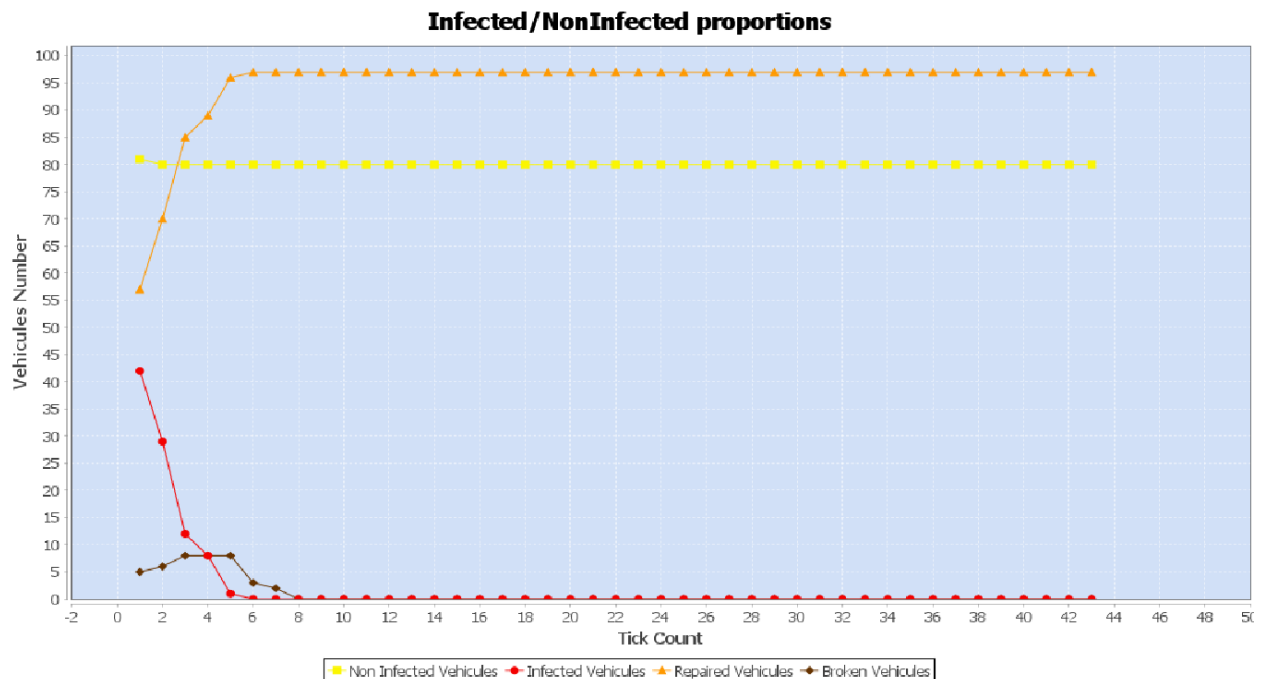
3. P_{rep} :

- Decrease $P_{rep} = 1\%$



Since an infected vehicle has a very low probability to get repaired, we can see that the number of repaired vehicles does not increase much. However, the number of infected vehicles takes more time to become equal to zero (Tick 46 here compared to 23 in the first version).

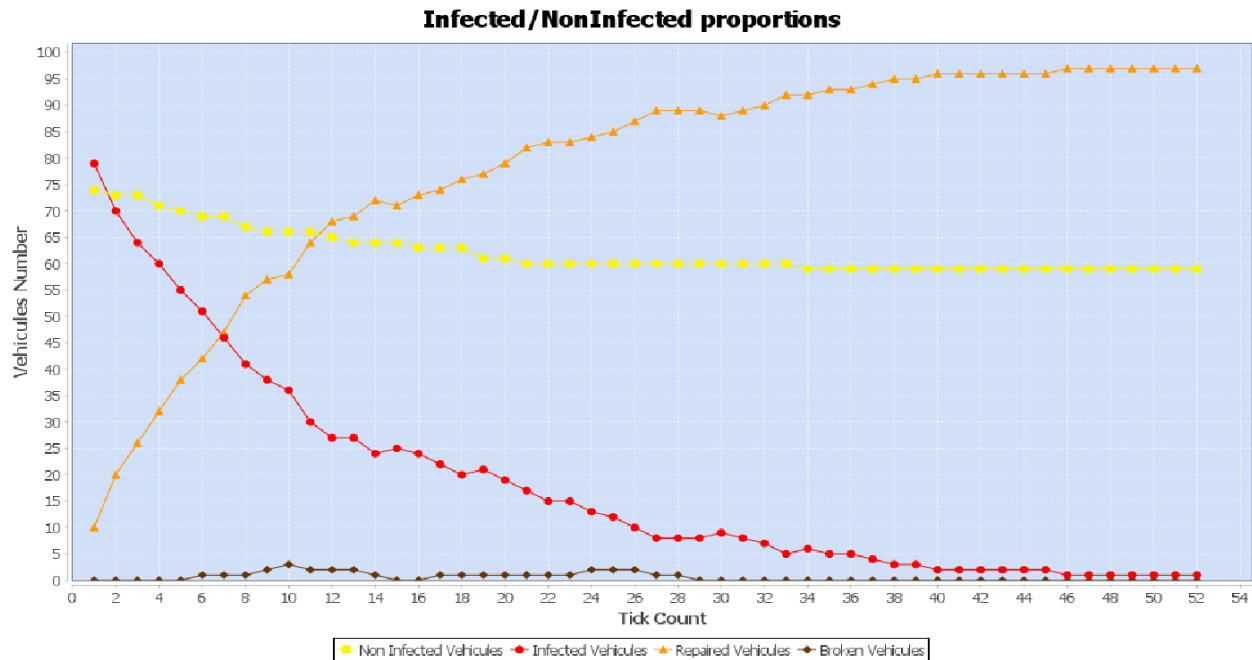
- Increase $P_{rep} = 60\%$



In this case, an infected vehicle has a 60% chance to get repaired, 10% chance to become broken and 30% chance to stay infected. Since the probability to get repaired is much higher, the number of infected vehicles decreases very quickly. Moreover, we can see very clearly that the number of infected vehicles and the number of repaired vehicles are inversely proportional.

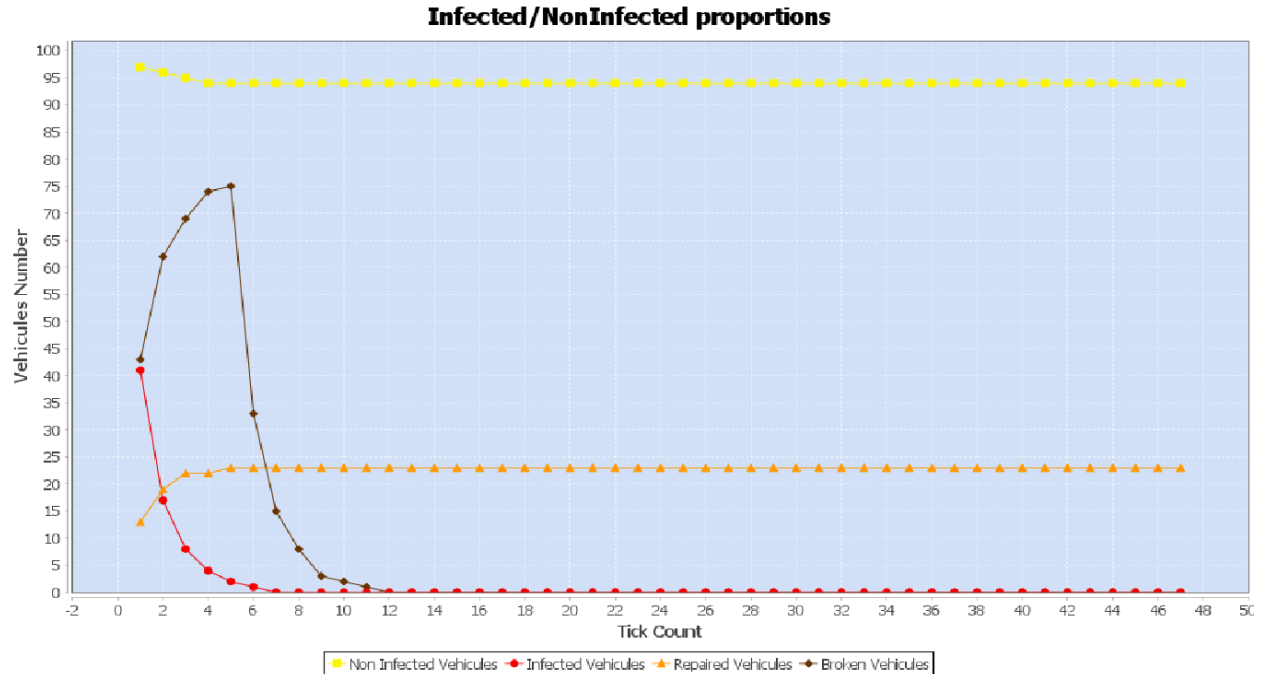
4. P_break :

- Decrease P_break = 1%



The broken vehicles experience an increase but slight as they have a less probability to get broken. Since vehicles take longer to get broken and removed, the number of infected vehicles takes more time to become equal to zero (Tick 46 here compared to 23 in the first version).

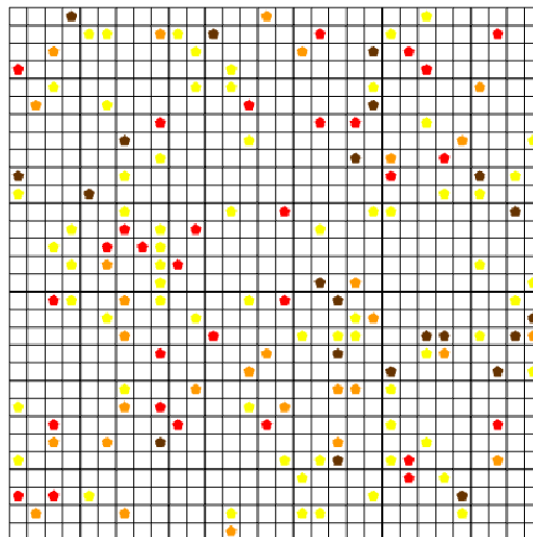
- Increase P_break = 60%

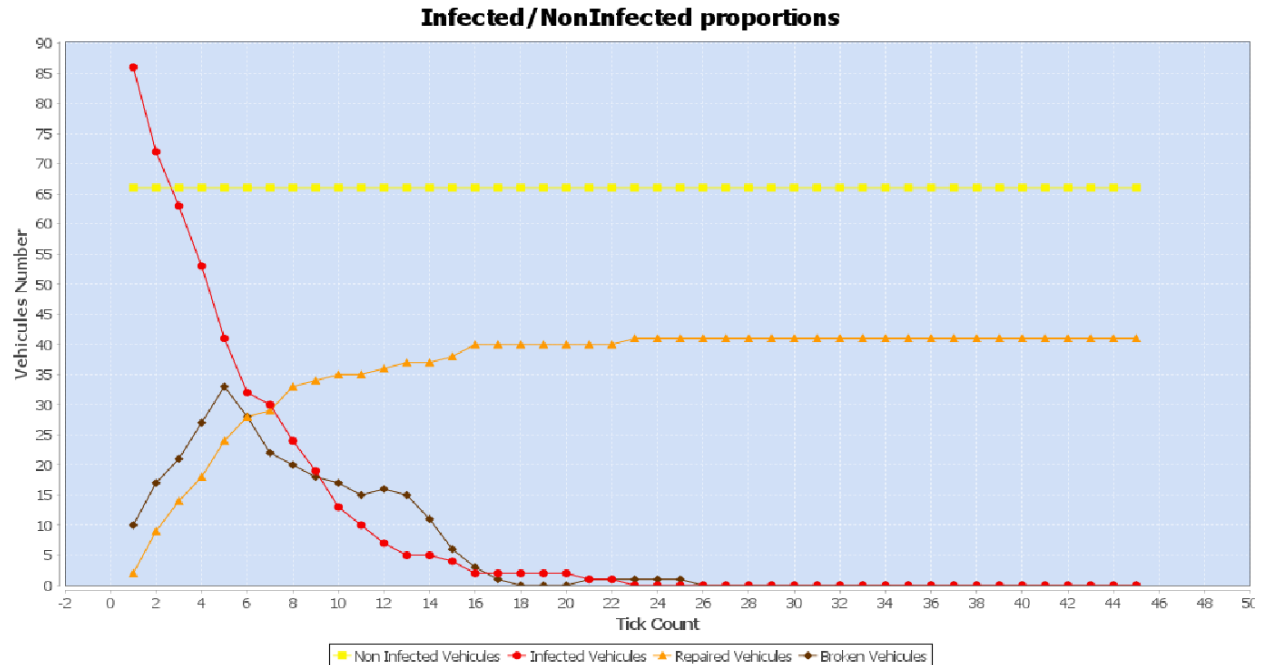


We can see very clearly that the number of infected vehicles and the number of broken vehicles are inversely proportional. Moreover, since the vehicles are more quickly broken and removed, the non infected vehicles do not decrease much.

5. Neighborhood structure :

- Remove the neighborhood structure





An non infected vehicle gets infected only in contact with infected vehicle. Vehicles meet only in public spaces. Since we removed the neighborhood structure, there are no more public spaces so vehicles don't meet, non infected vehicles don't get infected and for this reason, the number of non infected vehicles is constant.

5. Difficulties Encountered

The weakness of the project is the design of the district neighborhood which can not be changed. If we do not follow exactly the same type of design (Vertical roads + Horizontal roads + Blocks of 9 squares) we will lose the traffic mechanism. Although, it does not affect the mechanism of infection and the conditions on the type of space and the maximum number of vehicles in each type.

Indeed, making the design and implementing the movements of vehicles was the most difficult part of the project.

First, the coordinates of the grid are reversed. When you double-click on a grid element, the first coordinate displayed is the column, the second is the row number and the indexing starts from the bottom. It is not a big deal but it takes some time to get used to the (row, column) mode instead of the (column, row) mode.

Second, all the movements scenario choices depend on the design we defined at the beginning. And that's also the reason why we chose an easy design. The more we add shapes, the more complicated it is to adapt the movements of vehicles.

6. User Manual

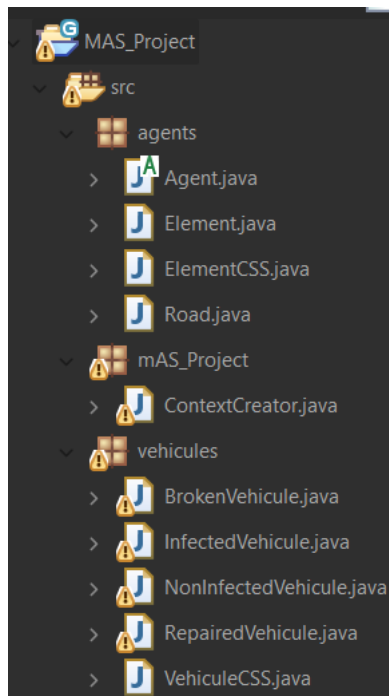
Requirements

- Repast Symphony installed
- Java 11 installed

Create project

1. File -> New -> Repast Symphony Project :
 - a. Give the name "MAS_Project" (without "") -> next
 - b. Libraries -> Add External JARs -> Path/to/java11 -> select java11
 - c. Finish
2. Under /src/ create 2 Packages :
 - a. agents
 - b. Vehicules
3. Under src/agents/ create 4 java classes and copy paste their code taken from the project source found with this report :
 - a. Agent.java
 - b. Element.java
 - c. ElementCSS.java
 - d. Road.java
4. Under src/vehicules/ create 5 java classes and copy paste their code taken from the project source found with this report :
 - a. BrokenVehicule.java
 - b. InfectedVehicule.java
 - c. NonInfectedVehicule.java
 - d. RepairedVehicule.java
 - e. VehiculeCSS.java

It should be looking like this :



5. Replace MAS_Project.rs/parameters.xml and context.xml by the files in the project source

6. Run MAS_Project Model

7. Data Loader -> set Data Loader -> Custom ContextBuilder Implementation -> MAS_Project.ContextCreator -> Finished

8. Displays -> Add display -> select grid -> next :

a. Select Element, Road, InfectedVehicule, NonInfectedVehicule, BrokenVehicule, RepairedVehicule -> next

b. In Agent Style section :

- Select agents.ElementCSS for Element
- Select vehicules.VehiculeCSS for InfectedVehicule, NonInfectedVehicule, BrokenVehicule
- For Road : Click on Edit agent style and select grey color and rectangle shape ->OK

c. Next -> Finish

9. Parameters -> Add Parameter -> gridHeight -> Default Value = 30 ->OK

->Add Parameter -> gridWidth -> Default Value = 30 ->OK

10. Run the program !



7. Conclusion

To sum up, we were able to simulate the system with the changes of the vehicles over time and according to the charts above, we can see how the vehicles get infected by time and how they get repaired or broken by time. We had challenges making the logic of the motion and make the system perform those changes but we were able to overcome it and find a way to work out the system and make it run at the end and make the simulation of the attacks.