FULLY AGENT BASED MODELLINGS OF EPIDEMIC SPREAD USING ANYLOGIC

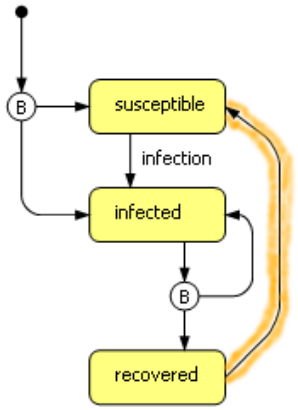
Štefan Emrich[1], Sergej Suslov[2], Florian Judex[1]

[1]Institute for Analysis and Scientific Computing,
Vienna University of Technology,
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria
[2]XJ Technologies Company,
Office 410, 49 Nepokorennykh pr.,
195220 St. Petersburg, Russian Federation

stefan.emrich@gmx.net (*Štefan Emrich*)

Abstract

The question how to model the spread of epidemics has been approached countless times. The number of different methods used on this problem is not too small either. Ordinary differential equations (ODE) and Partial differential equations (PDE) have dominated this field for several decades, if not centuries. In the second half of the last century two alternative techniques appeared on the stage, namely cellular automata (CA) and agent based (AB) models, also called multi agent systems (MAS). The difference between the approach with differential equations and the latter two methods is big. CA and MAS are so called "bottom-up" approaches, focusing on the smallest unit of the system – a cell or agent, whereas ODEs try to model the system via causal connections on the macroscopic level. Setting up a SIR-type model using the AB approach one can take advantage of state charts to control the behavior of agents. Using AnyLogic as implementation platform agents and especially state charts can be programmed very conveniently. Especially modifications and/or extensions of the final model can be handled in an elegant way. The right figure does show all necessary adjustments to expand the SIR- to a SIRS-type epidemic (additional state transition highlighted). The results obtained by simulation with such an MAS are comparable to those of the ODE- and CA-approach, although AB modeling offers a higher degree of freedom and thus more possibilities of adjustment.



Keywords: Epidemic Spread, SIR, SIRS, Agent-based modeling, AnyLogic.

Presenting Author's biography

Štefan Emrich was born on June 30th 1981 in Vienna, Austria. After finishing high school and his social service, he went on to study technical mathematics at the Vienna University of Technology. He became project assistant at the Institute for Analysis and Scientific Computing in February of 2007. He finished his studies 4 months later. Currently he is continuing his studies towards a PhD.

1 Introduction

1.1 History of Agent Based Systems (ABS)

The first known attempts to analyze patterns of epidemic outbreaks date back to Hippocrates [1]. Over the centuries the understanding of the biological and chemical processes leading to diseases became better and better, the mathematical knowledge grew larger and with it the possibilities and models for such attempts became more and more complex. At the beginning of the 20th century Kermack and McKendrick published the famous and often cited ODE-system

$$\begin{aligned} \dot{S}(t) &= -\beta S(t) I(t) \\ \dot{I}(t) &= \beta S(t) I(t) - \gamma I(t) \\ \dot{R}(t) &= -\gamma I(t) \end{aligned} \qquad (1)$$

for the simulation of a simple SIR-type epidemic, where *S(t)* denominates the number of susceptible individuals at time *t*, *I(t)* the number of infected and *R(t)* the number of recovered individuals. The variables β and γ represent the infection respectively the recovery rates. Since then the differential equation methods have been refined and improved in various ways. In the second half of the last century, with the rise of computers and increased computing power, cellular automata (CA) became an interesting field for experience. Especially one advantage made them quite interesting – the possibility to simulate heterogeneous populations. In the 1990ies computers finally reached a level making it possible to handle even more complex structures – ABS, also referred to as multi-agent-systems (MAS) were born. These systems allow for even more flexibility, as the agents may hold (many) different characteristics, such as age, or sex.

1.2 Characteristics of MAS

Cellular automata and multi-agent systems are both classified as "bottom-up" approach. Both methods describe (complex) systems by definition of local interactions. The interaction of CA is a very limited and strict one. Whereas AB-systems lack a clear and strict definition, but provide much more flexibility. Numerous varying definitions can be found throughout literature, with one possible definition (adapted from [2]) being the following:

An agent is a computer system situated in an environment, it has the capabilities to flexibly and autonomously act in this environment in order to reach its (predefined) objectives/goals. Which leaves three terms that need further specification:

- *Situated* in our case means that the agent is interacting with its environment. The agent is capable to receive input from the surrounding (generally via sensors) and can also manipulate it to some extent.
- The definition of *autonomy* needs to be handled with care, as we are talking about a pre-programmed computer system. Thus it is satisfactory if the agent can reach decisions without (human) interaction.
- *Flexibility* is required in multiple ways. Firstly one demands that the system is operating and acting in reasonable time. Secondly the agents are not to be solely reactive but goal-oriented or in the best case anticipating. And thirdly agents may have the capability to communicate or interact with other agents and/or real humans.

In MAS now several of such agents are interacting. Such multi-agent-systems are characterized by:

- agents have a limited point of view (incomplete information and/or problem solving capabilities),
- the absence of global system control,
- decentralized data and
- asynchronous computation of the agents.

2 Modeling

2.1 Simulation Environment – AnyLogic

AnyLogic is a programming and simulation environment, mainly aiming at modeling of hybrid systems, based on JAVA. It allows the user to combine different techniques and approaches such as differential equations, discrete events and agent based systems. These combination possibilities make it a very interesting tool for simulation of complex systems.

AnyLogic's newest version 6.0 is focusing mainly on the agent-based approach and business simulation, but allows other methods to be used. A drawback for the simulation of exact systems is the lack of a state event finder in the newest version. It is still available in version 5.5 but was dropped in 6.0 to improve runtime. On the other hand the performance of and possibilities for modeling of agent-based systems has largely increased. Numerous processes have been optimized and simplified.

A clear advantage of AnyLogic is the possibility to use JAVA code at any place of the program and thus expand or adopt the model to the programmers needs. Another neat feature of AnyLogic is the possibility to immediately create a JAVA applet of a working model, allowing it to be put on websites or distributed. In version 5.5 it took quite some time until one became accustomed with the program-layout, version 6.0 restructured the GUI, but still requires the user to get acquainted with it.

ISBN 978-3-901608-32-2

2.2 Definition of Task

The problem that we are going to conquer subsequently is a slight modification of the ARGESIM Comparison 17 (see [3]). This comparison does ask for the simulation of a SIR-type epidemic by means of lattice gas cellular automata (LGCA). At the end of this paper we will compare the outcome of such an approach with our ABS-result.

The task is to model a SIR-type epidemic, an epidemic simplified in several ways. For example we assume a constant population over the whole simulation, thus no births or deaths may occur. Further there is no incubation period or delay time between infection and infectivity of an agent.

CA are defined via their neighborhood, this means that we need an equivalent to the CA-neighborhood for our MAS. We are going to solve this by defining a field of view for our agents. This field of view is set via two parameters: vision-range or -distance and angle of vision (see Fig.1).

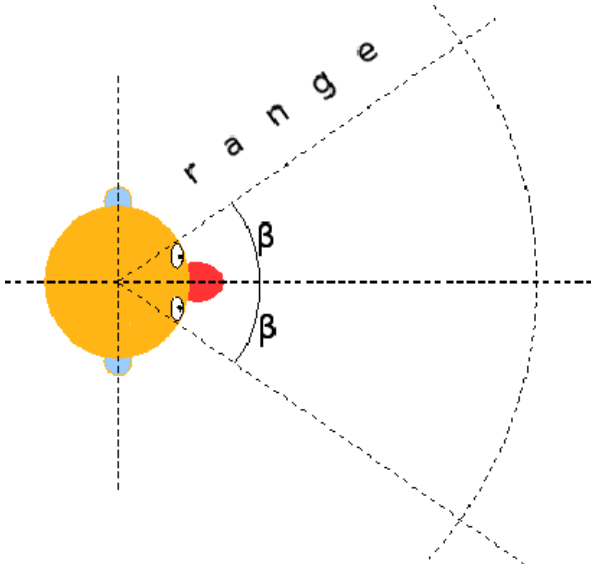


Fig.1 Field of View of Agent

The transmission of infection is being modeled as follows: if an agent crosses another agent's field of view they establish contact (for this every agent holds a Boolean control variable). In case that any of the agents is infected the infection is triggered probabilistically by a trigger-event.

Recovery of the agents is also reached stochastically, although with a given minimum for the duration of the disease. A recovered agent cannot become infected again, thus it is immune to the disease. This is leading to two possible scenarios:

- The disease infects all agents in our simulation resulting in a full immunization of our Population. The infection dies out because of lack of vectors
- The infection does not find any new vectors (e.g. the density of agents is too low) and thus cannot reach all individuals of our population before dying off. This leaves a partially immunized and partially susceptible population.

2.3 Model Set-up

As already stated in section 1.2, the basic element of an AB-model is the agent itself. Thus we start by creating an agent. This is done in AnyLogic by creating a new class. The agent is assigned 5 variables, two real variables for the X- and Y-coordinate of the agent and another two real variables for the X- and Y-component of its movement. All four of these variables are random numbers with an upper limit - either a given maximum speed or the boundaries of the simulation environment. The fifth variable is, as said before, Boolean and in charge of controlling the contact status of our agent.

The health status of our agent is controlled by a state-chart which represents the possible states and defines the state-transitions and actions for these transitions (see Fig. 2 for the health state chart).

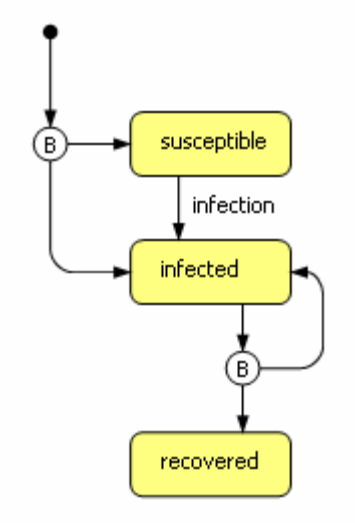


Fig.2 State chart "health" of agent

The circled B's in the chart represent branches where (in our case) probability comes into play. By this we also model the initial infection of the population.

Next we need to define the state chart for movement and contact of our agent. This is accomplished by the state chart presented in Fig. 3.

ISBN 978-3-901608-32-2

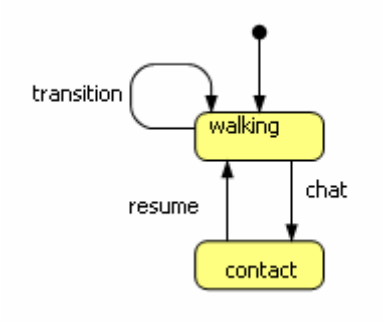


Fig. 3 State chart "movement" of agent

The transitions within the state charts need to be triggered by certain events. An example is the procedure of infection that was already mentioned in section 2.2. On contact the health states of both agents are checked for an infected. If an agent is infected and its contact susceptible, a message is sent to the contact (with given infection probability). If such a message is sent, it triggers the state transition from susceptible to infected.

A transition (within the state chart movement) is triggered in case that an agent reaches the border of the simulation environment. In this case several options are possible. Since our population must stay constant we can either reflect the agent into the environment (by simply switching its direction) or we can periodically insert it on the opposite side of the environment.

In AnyLogic it is also possible to add visualization to every class. This allows specifying a shape for out agent which can range from a simple dot to a complex object. We can keep it simple and just use colored dots that change color according to the status of the agent.

In a next step we replicate our object "agent" as often as required. This is very well supported by AnyLogic and the agents can be placed easily in the main simulation. The overall simulation visualization is also located in this main environment, together with control elements. These can be slide bars to change certain parameters during runtime, graphs that show the current number of infected individuals, etc.

3 Experiments

3.1 Variation of Movement Speed

First we will take a look on how variation of the maximum speed affects our model. As implied before, the speed of the agents is not the same, but assigned randomly within a certain interval from 0 to maxSpeed.

Figures 4, 5 and 6 do show the variation of the maxSpeed from 10 to 30 and 60 units. (The vision range is kept at 15 during all experiments of this first set, allowing better comparability with the second set of experiments).

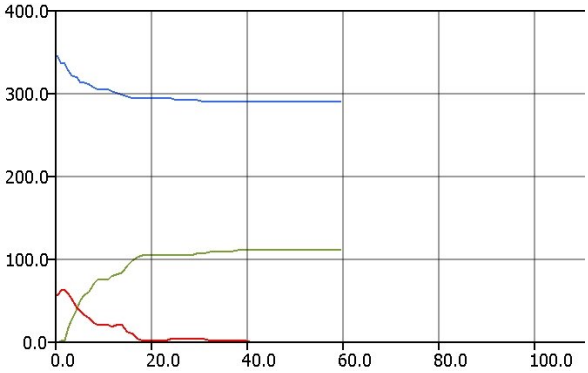


Fig. 4 maxSpeed of Agents set to 10
(Figure corresponds to Fig. 8)

One can easily see the difference between the runs if looking at the subpopulation of infected agents. The outcome becomes even more obvious if one is observing the subpopulations of the susceptible or immune agents and the differences between the graphics.

We further notice, that all three settings of the model lead to the case we mentioned before, in which the infection dies out before reaching all individuals of our population.

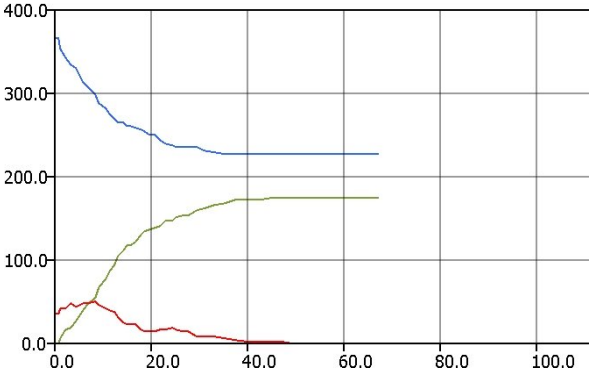


Fig. 5 maxSpeed of Agents set to 30

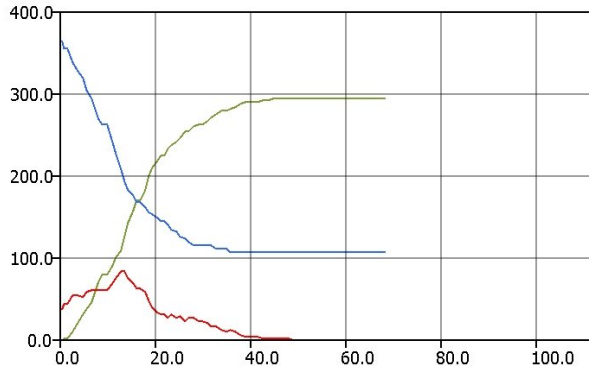


Fig. 6 maxSpeed of Agents set to 60

3.2 Variation of Size of Vision Field

In a next step we will alter the size of the vision field of the agents. If we take another look at Fig. 1, we can easily see that this can be done in two different ways. We may either in-/decrease the distance or the angle of the vision field. In either way we will reach the same result as the area of the field is in-/decreased and thus the probability that another agent passes trough it. The shape of the vision field is of no importance.

We choose to modify the vision distance from 5 to 15 and 25 units. For better comparability with the first set of experiments the maxSpeed of the agents is set to 10 throughout this set. Thus Fig. 8 corresponds to Fig. 4 of the first set of experiments.

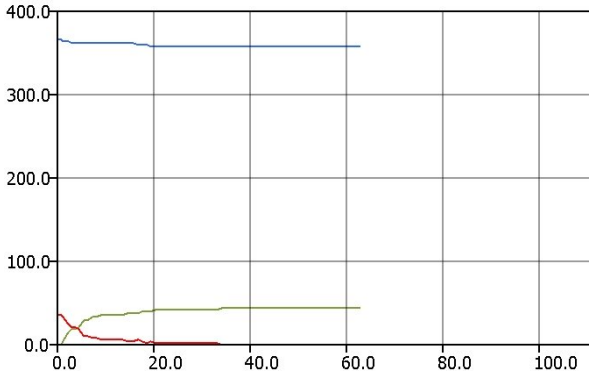


Fig. 7 Vision range of agents set to 5

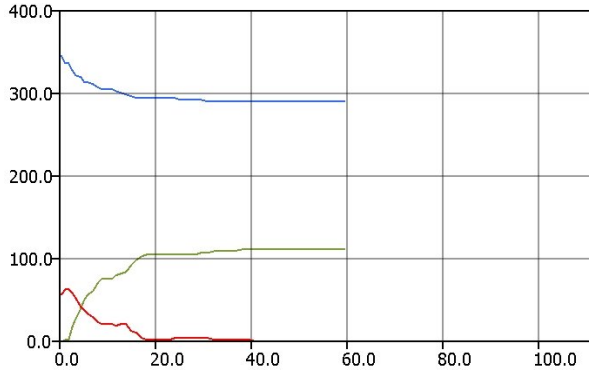


Fig. 8 Vision range of agents set to 15
(Comparable to Fig. 4)

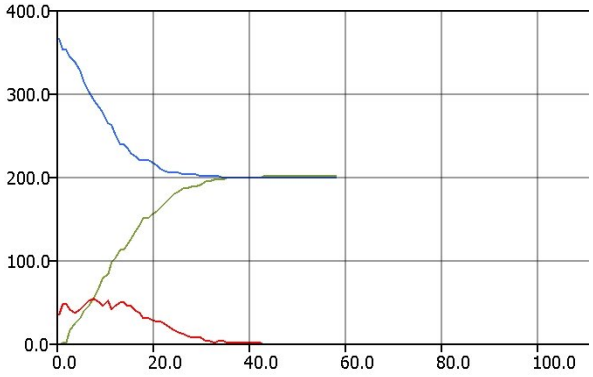


Fig. 9 Vision range of agents set to 25

This set of experiments presents similar results as the ones obtained from simulations in the first set. This is not further surprising, as both parameter variations are leading to a higher contact rate between the agents in the system.

The infection and recovery rates have not been changed through all of these experiments. By modification and parameterization of all available parameters one can adjust the model to fit the curves produced by the Kermack and McKendrick ODE-system presented in Eq. (1). Although in general the infection and recovery rates will not be the same.

3.3 Expansion of Model – SIRS

Now we will expand our SIR-type model to simulate a SIRS-type epidemic, this epidemic pattern is characterized by the fact that recovered (immunized) agents do become susceptible again, allowing them to get infected repeatedly. This changes the whole behavior of the system, as it adds the possibility, that a disease does not get instinct but becomes endemic within the population.

In our case the modification necessary to achieve such a model behavior is very simple, the implementation in AnyLogic fast and convenient.

We remember, the state chart "health" (see Fig. 2) of our agents that controls the health status and now return to this state chart to add another transition, which takes just three clicks, and modifies our state chart as follows (see Fig. 10).

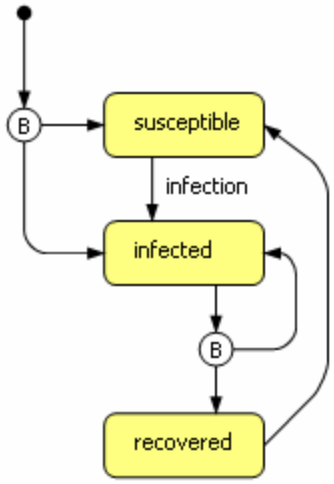


Fig. 10 Modified state chart "health of the agents. Notice the additional transition on the far right from recovered to susceptible.

Of course we can implement several triggers for the recovery transition. This for example can be done either stochastically, such as infection and recovery or by timeouts or a combination of both. We will use a fixed timeout in order to obtain "nice" results.

ISBN 978-3-901608-32-2

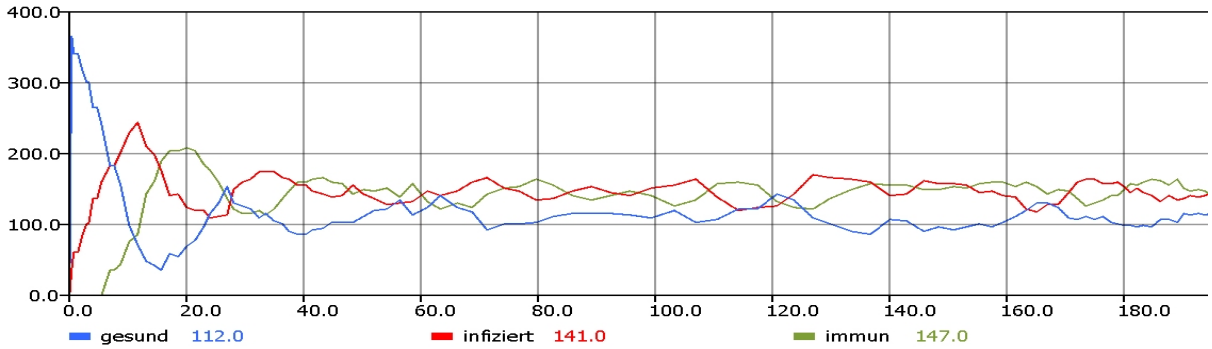


Fig.11 Duration of immunity set to 10 units of time

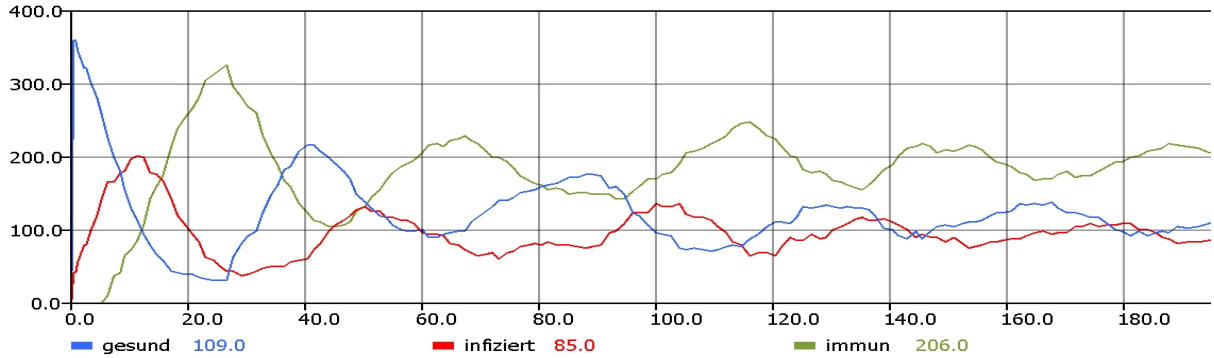


Fig.12 Duration of immunity set to 20 units of time

In the subsequent set of experiments we will change the duration of the immune period, starting from 10 units of time and raising it to 50 units by steps of 10.

In such a modified system one notices the earlier mentioned behavior – the disease becomes endemic and does not die out, but keeps at a more or less constant level (see Fig. 11).

If we now increase this parameter we will notice a nice effect – the epidemic will come in waves (see Fig. 12, 13 and 14).

These experiments show very clearly that with increasing length of immunity the frequency of the epidemic waves decreases.

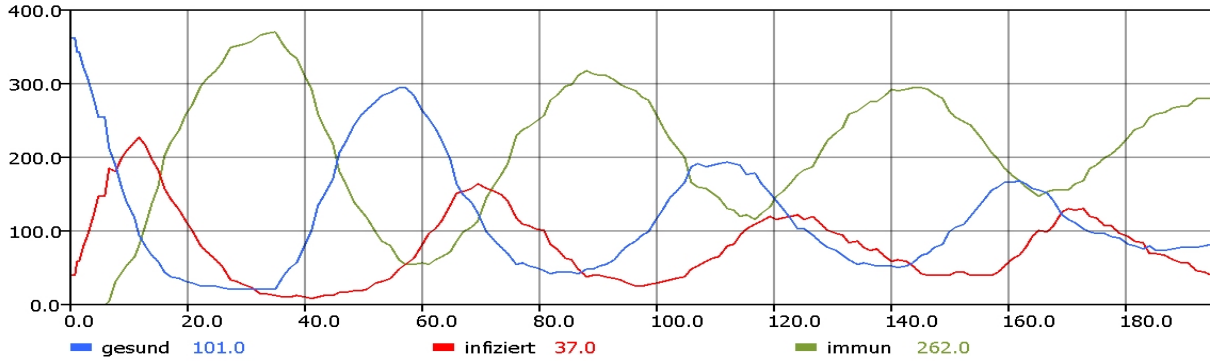


Fig.13 Duration of immunity set to 30 units of time

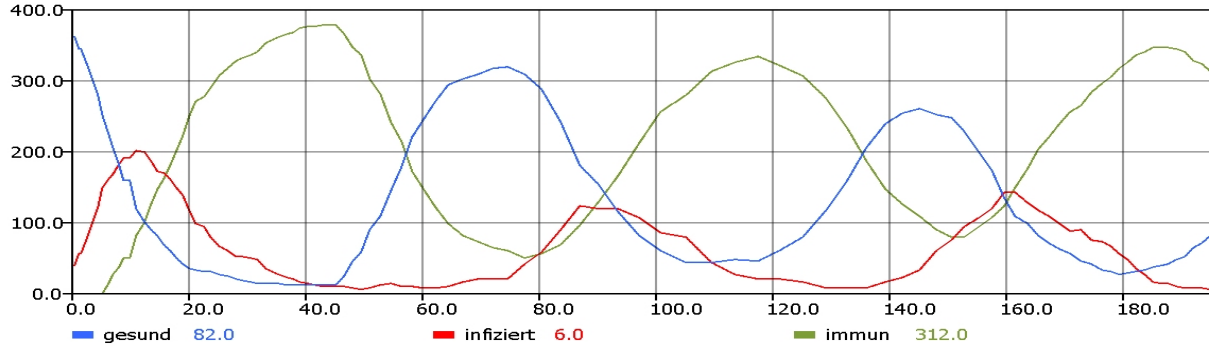


Fig.14 Duration of immunity set to 40 units of time

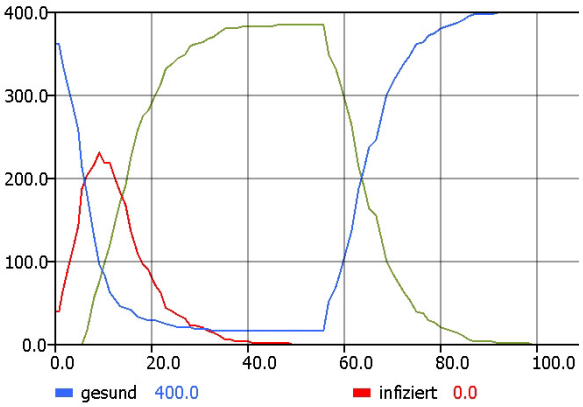


Fig. 15 Duration of immunity set to 40 units of time

In Fig. 14 the curve of the infected population already almost reaches the lower bound. If this happened there would be no more infected individuals within the systems meaning that the infection would have died off. This is the case if we further increase the duration of immunity (see Fig. 15). The epidemic dies out and after a delay (the duration of immunity) the whole population becomes susceptible again.

If we compare the epidemic patterns to results obtained using a lattice gas cellular automata (LGCA) for the modeling of a SIRS-type epidemic, we can see that the qualitative behavior is fairly similar (see Fig. 16).

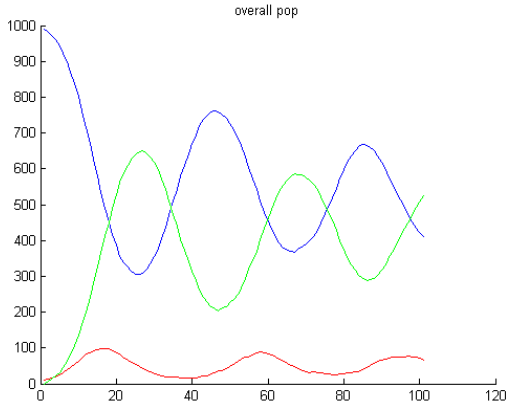


Fig. 16 Simulation of SIRS-type epidemic using LGCA (implementation in MATLAB)

The smoothness of curves in Fig.16 is explained by the fact that the graph represents the average obtained in 40 consecutive simulation runs (Monte Carlo method).

Of course the quantitative behavior of the models differs, but this again can be solved by a proper parameterization of the model(s).

4 Conclusion

This simulation of a SIR-type epidemic using an agent-based approach in AnyLogic shows some quite interesting things. One notices, that the qualitative outcome of the model corresponds to expected system behavior known from other modeling techniques, such as ODE-systems or CA-approaches. Thus using the agent-based approach appears to be legitimate – at least in this case.

Further AnyLogic proved to be a sufficient platform for the modeling of MAS as it provides effective and helpful tools to the programmer. Especially the implementation of state charts and state events is very convenient. On the other hand it also showed some short comings, especially in terms of data post-processing. It does not offer a built in functionality to store the data obtained during simulations. Since AnyLogic is based on JAVA this can be conquered by a JAVA-based work around, e.g. exporting data to a spreadsheet or csv-file during runtime. Although this work around needs to be implemented manually.

A big advantage of the agent-based approach was shown in section 3.3 when the model was expanded. Additions or modifications of the model can be implemented very easily by simply adding the desired features to the agent.

Although major improvements in terms of runtime have been achieved from AnyLogic 5.5 to version 6.0 a weakness still remains when simulating huge systems. But altogether simulation of systems using MAS appears a convenient and effective method for certain applications, especially for socio-economic problems where many aspects and interactions need to be considered.

5 References

[1] Baylei Norman T. J. *The Mathematical Theory of Infectious Diseases and its Applications.* Charles Griffing & Company LTD, 1975.

[2] Jennings, Sycara, Wooldridge. A Roadmap of Agent Research and Development (Autonomous Agents and Multi-Agent Systems). Kluwer Academic Publishers. 1998

[3] H. Hötzendorfer, N. Popper, F. Breitenecker. *Temporal and Spatial Evolution of a SIR-type Epidemic – ARGESIM Comparison C17 – Definition*
http://www.argesim.org/comparisons/index.html

