UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

"VACCINES VILLAGE"

NETLOGO SIMULATOR

Student: Marina Vallejo Vallés
Professor: Dr. Marta Ginovart Gisbert
Subject: Programming and Problem Solving in Engineering
Bachelor's degree in Biosystems Engineering (UPC)
05/06/2020

INDEX

1. INTRODUCTION

To carry out the project of the subject "*Programming and Problem Solving in Engineering*" I chose to build a NetLogo simulator from scratch.

The simulator named "VILLAGE VACCINES" recreates the arrival of a traveller to a village. This traveller has been visiting a country where there is a disease that is already eradicated in his village. But as the traveller is not vaccinated, he ends being infected.

The simulation begins with the arrival of this traveller to the village. He will infect some unvaccinated residents he meets. Note that not every resident he meets will become infected. In order to become infected, first of all the villager should not be vaccinated. Then a random number is generated in order to determine if he becomes infected or not. With this approach we will obtain different results every time we run a simulation.

Those **infected villagers**, will be spreading the disease to others. They can be differentiated from the not infected villagers by the colour, since they will present a **purple colour** in the "world".

It is also possible to see the temporal evolution of different variables such as the total population, population stratified by gender, number of infected residents and the number of deaths. There are several monitors and a graph where the user can check the status of these variables.

The simulator aims to make the user aware of the consequences of the population not being vaccinated. Today, the anti-vaccine movement is gaining considerable strength in society. The fact that a percentage of the population is not vaccinated can cause diseases that are already eradicated in most territories, may have a resurgence and this would be an important health problem.

In Figure 1 we can see the simulator interface, with the default start parameters.

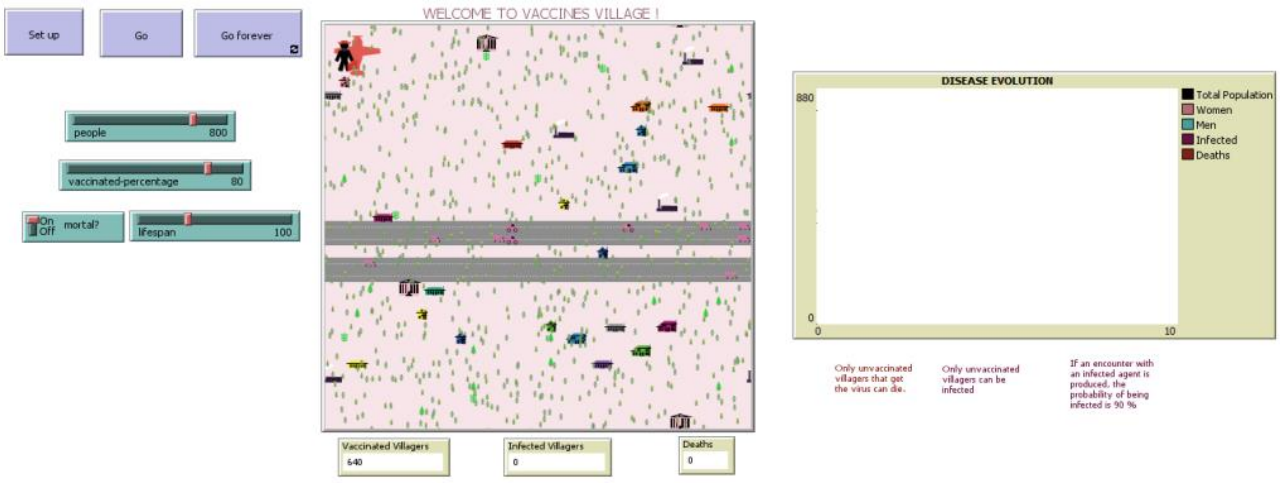

Figure 1. Vaccines Village simulator interface.

2. ODD PROTOCOL

Here it is detailed the **ODD Protocol** of the simulator "VACCINES VILLAGE". We describe in a structured way the simulator created in the present work.

- **O: Overview**

Purpose. To show the user the consequences it would have on a certain population if a percentage of people were not vaccinated and there was an outbreak of an already eradicated disease in that population.

State variables and scales. The model is composed of **mobile entities** (turtles) and **immobile entities** (patches). The turtles can be differentiated as the traveller (called "visitor" in the code) arriving by plane and the residents of the village (called "villagers").

We control the following variables for the turtles:

I. **Cartesian coordinates** of its position in space, or "world". Determines the patch on which the turtle is located.
II. **vaccine?**: Boolean variable which is TRUE if the individual is vaccinated. For the traveller, this variable will be always FALSE.
III. **infected?**: Boolean variable which is TRUE if the individual presents the disease and is infectious, and FALSE in case of being a non-infectious healthy individual. Those vaccinated villagers will never be able to have this variable with value TRUE. For the traveller, this variable is always TRUE from the beginning.
IV. **time-to-death**: number of remaining days until the infected individual dies.
V. **arrival?**: only defined for the traveller. Boolean variable which is FALSE in case of the traveller not having arrived to coordinates (1,3).

Each temporal unit will be equivalent to a **tick**. We will consider ticks as weeks.

Process perspective and programming. Time is discretized and controlled with **ticks** (program steps).

Behaviour rules that apply to individuals at every step of the program are:

I. Movement directed towards the xy coordinates (1,3). Only for the traveller.
II. Random movement within the xy coordinates present in the "world".
III. If it is an infected individual, decrease by a unit the remaining time of life at each program step.
IV. If it is an infected individual, in case of matching in the same patch with non-vaccinated villagers, generate a random number to assess if he will infect or not. Considering an infectivity of 90%, infect susceptible individuals who are in this patch.
V. In the case of a vaccinated individual, he or she cannot be infected.

Non-infected individuals are represented in **green**. Whereas **infected individuals** are **purple**.

There is no colour difference that allows us to know if the individuals are vaccinated or no.

This model does not contemplate the birth of new individuals or deaths by other causes unrelated to the disease studied.

- **D: Design Concepts**

This step is not contemplated during the present work.

- **D: Details**

Initialization: Table 1 shows the different parameters that are presented in the interface, where the user can modify its value. Apart from these parameters the user will also find the input variable "**mortal?**", where its default value is "**ON**". This variable allows us to decide if the disease is fatal or not. If the user switches the variable to "**OFF**", the parameter "**lifespan**" will not be assessed by the code.

VARIABLE	*DEFAULT VALUE*	*MINIMUM VALUE*	*MAXIMUM VALUE*
people (Number of initial residents)	800	100	1000
vaccinated-percentage (Percentage of vaccinated residents)	80	0	100
lifespan (Remaining days until death once you are infected)	100	50	200

Table 1. Initial parameters for the simulator Vaccines Village.

Entry: This model doesn't take into account migration movements.

Submodules:

move-cars
`to move-cars`
`  ask cars [ set heading 90 fd 1 ]`
`end`
	The cars in the simulator look to the right and will move a patch at every step of time. They are only used for decorative purposes.

move-planes
`to move-planes`
`  ask  planes [ facexy 1 3 lt 4 fd 1`
`  if (pxcor = 1) and (pycor = 3) [die]`
`  ]`
`end`
	The plane will head towards the coordinates (1,3) and once it arrives it will disappear from the model since it is not relevant. We only use it to represent the arrival of the traveller.

move-visitor
`to move-visitor`
`  ask visitors [ ifelse arrival?  [ rt random 100 lt random 100 fd 1 infect`
`  if mortal? = TRUE [set time-to-death time-to-death - 1`
`  if time-to-death = 0 [ set number-dead  number-dead + 1  die ] ]]`
`  [ facexy 1 3 lt 4 fd 1 if (pxcor = 1) and (pycor = 3) [ set arrival? true ]`
`  ]`
`  ]`
`  stop`
`end`
	The traveller is heading towards the coordinates (1,3) as the plane. Once it arrives, it changes **arrival?** to TRUE. From this moment he will move randomly and will be able to infect the unvaccinated villagers he finds (as long as the draw allows it). In case that the disease is fatal, in each time step it will decrease in a unit its lifespan. When it is equal to 0, he will die.

things-villagers
`to things-villagers`
`  ask villagers [ rt random 100 lt random 100 fd 1`
`  if infected? = TRUE [ infect`
`  if mortal? = TRUE [ set time-to-death time-to-death - 1`
`  if time-to-death = 0 [ set number-dead  number-dead + 1  die ]`
`  ]`
`  ]`
`  ]`
`end`
	The villagers move randomly. In case of being infected, they will be able to infect other villagers. If the disease is fatal, in each time step it will decrease in a unit its lifespan. When it is equal to 0 the villager will die (as the visitor).

infect
`to infect`
`  ask other turtles-here with [breed = villagers and not vaccine? ]`
`  [`
`  if random-float 100 < 90 [ set infected? true  set color magenta - 1 ]`
`  ]`
`end`
	The traveller and those infected villagers, at every step of time enter this subroutine. If other villagers are found in the same patch in which they are and these are not vaccinated, a draw is made randomly. If the number obtained is less than 90, they will infect those villagers and will change their colour to magenta, in order to differentiate them from healthy uninfected individuals.

Table 2. Subroutines taken into account in the simulator Vaccines Village.