An Epidemiology Model on Netlogo

New Mexico

Super Computing Challenge

Final Report

April 1 2015

Team Number

Las Cruces High School

Team Members:

Jonathan Baca

Selema Graham

Teacher:

Elisa Cundiff

Project Mentor:

_____

Executive Summary

We completed an epidemiology model using Netlogo to show how illnesses can spread (and mutate). This model can show what is needed for an illness to wipe out a population, possibly the world, it also only shows one type of the virus—it is one that doesn't mutate. Said model can also show almost any illness. The user can change very many variables, such as the population, how many immune people- if any, mortality rate (but the higher the mortality rate in the program the longer it takes for them to die), how many hospitals, the space needed (radius) that another person needs to be in to become infected, and how likely it is that someone it is going to leave an patch of disease for someone else to step on and become infected. The user can also decide if the illness already has a cure vaccine that is available to the public, or if the disease can be acquired again. By implementing and changing these variables the model can show many different illnesses, and also if it can have the right numbers to wipe out the world. This could prove useful in terms of determining which variables would create a devastating disease, so that we can better prepare or identify these pathogens. This model also easily can test if the number and placement of hospitals can affect the spread of the disease, as well as if having a vaccine for it can help reduce the spread. Moreover, the model can show if the number of immune people affects the rate at which the specific disease spreads. We would like to be able to show the accuracy of the model by inputting an actual disease's numbers and seeing if it has an accurate representation of that illness.

Introduction

The computer program we are making is called the “perfect disease”. We are building this program on Net Logo because it is agent based modeling and is easy to manipulate to get all the different variables in. Including how the user can change very many variables just in the interface. Because of this, modeling just one would be predictable and unhelpful, but to be able to model just about any illness that you had the variables for could be helpful and unpredictable. That is why we chose to use Netlogo. This computer simulation is on the topic of epidemiology. We are trying to simulate the “perfect disease” as well as many others. Epidemiology is defined as the branch of medicine with the incidence, distribution, and possible control of diseases and other factors relating to health. As we were creating this program we learned how random diseases can be. Plus, there are very many human diseases in the world, no one knows quite how many for sure but it is normally estimated in the low hundred thousands. Diseases have so many factors and uncertainties that dealing with them can be hard. Epidemiology is something that many people use to try and get answers about many illnesses and diseases. Some branches include trying to find cures, studying how it mutates, looking at chance of distribution after a natural disaster, trying to keep the infection under control, going out into the place that an outbreak has occurred and trying to find out many things about it, et cetera. People who deal in some of these fields are possibly risking their lives, and for a fact saving lots. Dealing with diseases (that can be lethal) coming up with explanations, and answers to tell governments about how to handle a pandemic emergency; all these are just a few examples of what Epidemiology is, what it is about and what they do.

Problem

The problem we decided to solve was what kind of disease would be needed to infect, and wipe out the world. We were then curious to learn about other diseases, and see if we could make it general enough to model pretty much any disease. We focused on those two problems: trying to make a program with enough variables so that it could model pretty much any infectious pathogen, and also had the capability to infect and wipe out the world.

Solving

To get to the solution we had to do a lot of research, like finding out what makes pathogens different from one another such as: infection rate, the amount of people immune, how much space is needed to become infected (such as a spread through contact, or more like sneezing), if there is currently a vaccine available to the public, if the pathogen is reoccurring (as an example, chicken-pox is not, but most influenzas are), and the mortality rate. It also seemed necessary to add in how many hospitals there are, and be able to choose the number of immune people. While coding there were many bugs that we had to work out, and solve. We solved this by trying new things and re-coding also, with some help from similar Netlogo programs.

Validation

To test the program, we inputted many different parameters to see what the program did. It gave many different results some had the world pretty much dead, and some had pretty much everyone get immune, while others seemed to run for a while with the sick people and healthy people bouncing back and forwards—much like an ecosystem. Most of the runs depended on if there was an available vaccine or not. That changed a lot of the outcome, and in fact did keep the infection rates down, as would be expected, because implementing it into the real world if vaccines didn't slow infection, and keep people (pretty) safe from illnesses we wouldn't bother getting/making them. Also how much the immune people can help slow infection, because (even without vaccination) people naturally immune can't get infected so, there are less sick people around the healthy people to get them sick. This applies in life because if a family is all vaccinated but they have a baby who isn't old enough to get a vaccine, then the family not getting sick will keep the baby from getting sick. This is a great way to think about the program, instead of individual people, imagine each "turtle" (as they are called) as a family, but abstract the part where they go away from their families. Another variable that had a pretty big impact was the amount of hospitals, because they helped make people healthy again—in other words, cured them. It changed a lot because without them, people just got infected and then soon everyone (except the immune people) was sick and not getting better, then—depending on the mortality rate—the people died. But when there were a lot of hospitals the people would easily walk into one and either become immune or healthy again, or stop by for a visit (also known as they were already healthy, but there wasn't a vaccine). Something that was noticed was the how much more the pathogen could spread once the agents started leaving patches of the pathogen around. This simulates the real-world because if a sick person were to sneeze or cough into their hand and then touch a door-knob, it

would leave the pathogen on the door-knob waiting to be picked up. If a healthy person came into contact with a pathogen like this, they are almost certainly going to get sick, if precautions aren't taken. For simplifying reasons, in the model they get sick immediately. Also in the model, if an immune or already sick person were to come in contact with an infected patch, it would just disappear. This applies to reality because obviously an immune person isn't going to get sick, and an already sick person might be affected, but they are still sick.

Results

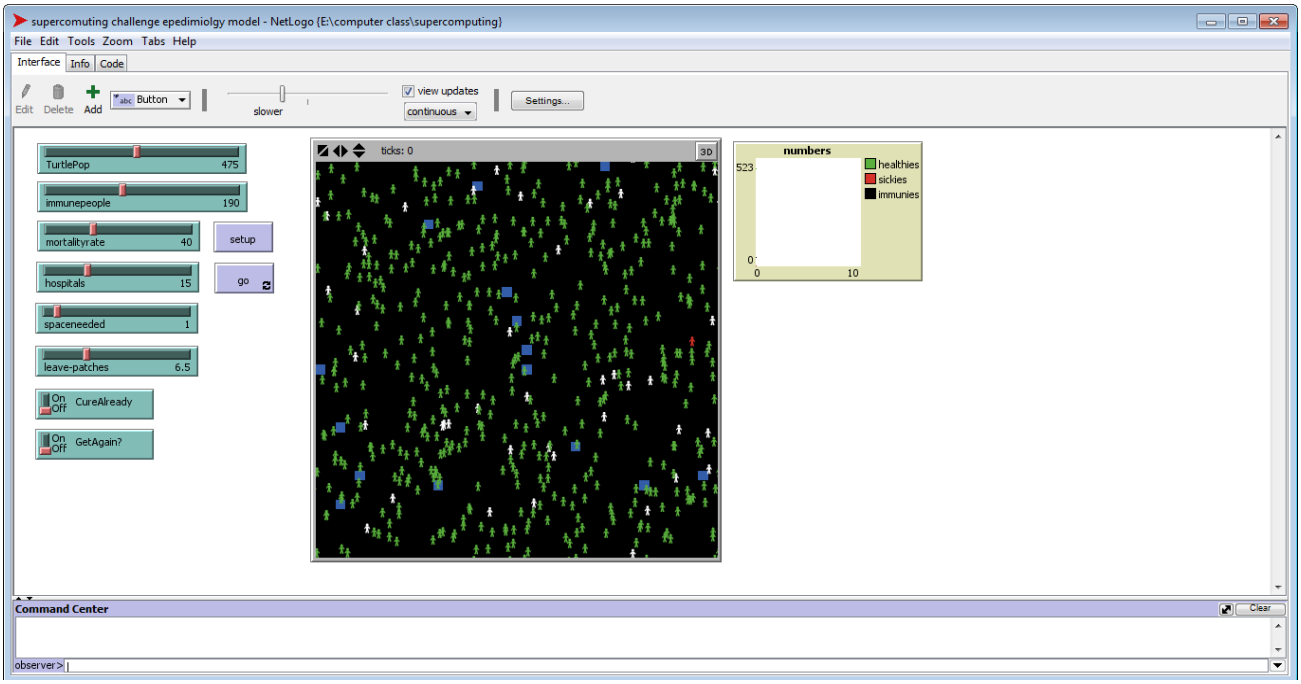
Our study showed us about what is needed to make not only a "perfect disease" but any disease in general. In order to wipe out the world it was learned that there couldn't be any immune people, and the mortality rate had to be high enough that it could spread before they died off, and they don't have a lot of time to get cured—in the model it was between 30 and 40 (unless you want to wait awhile, which gives them time to go to a hospital). It is a lot more likely to happen if they left patches, in the model 6.0 was good to use. The radius to be in to get infected was fine at 1, there was no cure, and they could get infected more than once, and around 15 hospitals to keep it seeming real and still be able wipeout the population. These numbers came pretty close to killing the whole world. By the time all the sick people had died there were a few people (less than ten) and that isn't enough to start up again. Even to put a model to world ratio, it would show that there is a possibility of there being an infectious disease that can wipe out the world.

Conclusion

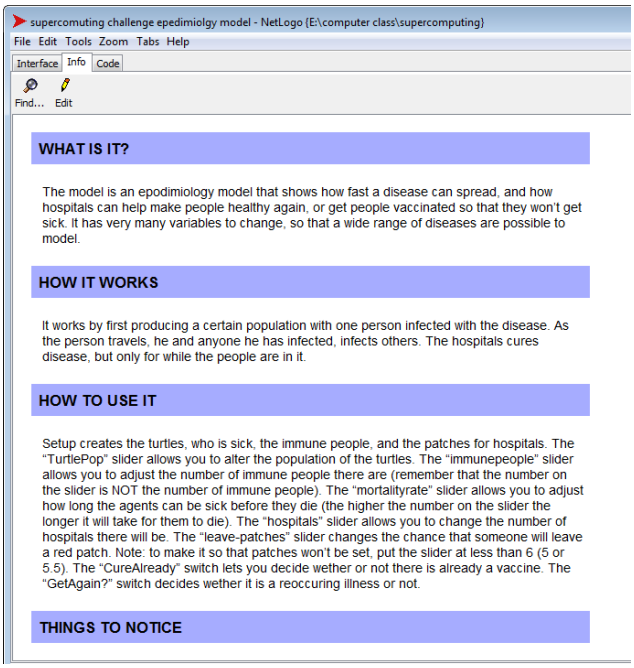
From our study we have learned that not only is there the possibility of a pathogen capable of wiping out the world, it just needs to be quick enough that we can't develop a cure in time. We also realized what it takes to make a devastating pathogen, which could be: smart mortality rate, not super good hygiene, no vaccine/long time to get one, et cetera. Models like this help the world because they help epidemiologists understand more and experiment with pathogens, without necessarily risking theirs or others' lives. It also helps them get an understanding of the pathogen they are dealing with in a very life-like way, but not using human experimentation, and it is also a lot quicker than waiting and studying actual people for days, months, or years. We also learned how very many things can affect an illness, even in the slightest way, which makes modeling epidemiology more difficult, and also makes each program and programmer different in what they choose to include.

Thank you to, our teacher Ms. Cundiff, for her help and support and deadline reminders. Then there is Sam Reed, who was my partner in class, and helped with the very first part of the coding, which was just to get a working virus. Also, the very nice people at the Kick-Off Conference, and the project evaluations, who helped direct us in where to go with our project, with their constructive criticism and helpful praise.

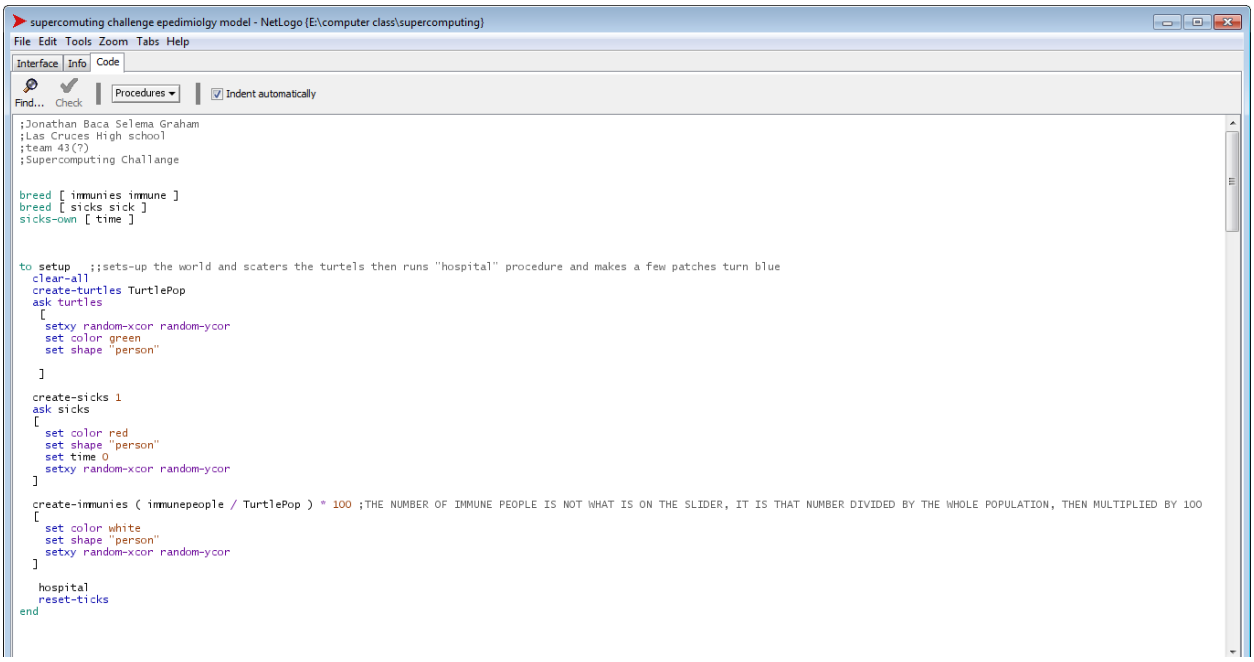
We feel our biggest achievement was not only being able to get together and get something like this done, but also being able to program the whole thing, and report on it. The sense of achievement we feel looking at this program run, and reviewing the code, is a great feeling that we wish to have many more times in our life.



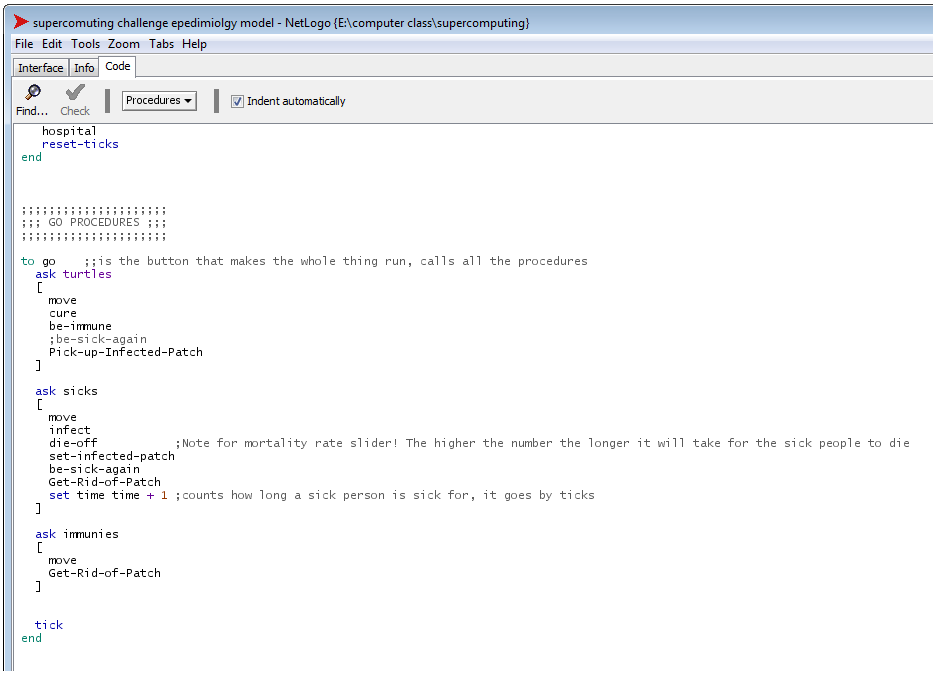
The interface, before hitting go.



Don't forget to take a look at the "Info" tab.



The setup procedure, also the header and some breed declarations, and an “agent-variable”.



Go procedures.