Disease Spreading Model using Probabilistic CA

Diplav D. Dongre
M.Tech I Year
Visvesaraya National Institute
Of Technology
Nagpur, India

Gourav J. Sharma
M.Tech I Year
Visvesaraya National Institute
Of Technology
Nagpur, India

Jayshil H. Patel
M.Tech I Year
Visvesaraya National Institute
Of Technology
Nagpur, India

ABSTRACT
In the computational era Cellular Automata is one of the best tool for performing the complex computation at the high speed. In cellular automata most of the complex computational problems from various fields are modeled graphically. In this article we tried to explore the CA functionalities and some of the problems exist in the basic CA model like glider. We have implemented and simulated a 2-D basic disease spreading problem using basic CA with the help of NetLogo tool.

Keywords
Cellular Automata (CA), epidemic model, disease spreading, glitter, probabilistic CA model

1. INTRODUCTION
Most of the biological problems have characteristics of randomness. To understand such problems we need a computational model which has the capabilities to model structure of such a problem. The epidemic model [2] of disease spreading [3] is a real world model. Such a real world model does not follow a defined structure like we don't know which person will have the disease and where he will travel and disease will spread.

Modeling of the above stated problem needs a computational model which has capabilities to process huge data and simulate the real world scenarios. Such a computational model also needs to feedback the previous state of the data to next state in the model.

Cellular machines are computational model, first studied and examined by John Von Neumann in 1940. The main goal of CA was to design Self Replication System that was supposed to be perfect computationally.

Cellular Automata is dynamically evolving system that act as a good model for physical, biological, sociological phenomena and help to solve computational problems. CA is used as a model for traffic and urban growth. Also, CA has key applications in commercial computer graphics, simulation of biological systems and to the design of massively parallel computers.

A standard Cellular Automaton is an array of cells, each can be in one of a finite number of possible states i.e. a 0 or a 1 (Boolean automata). The regular cellular array (lattice) is of 1-Dimensional, 2-Dimensional or 3-Dimensional. In a CA, the cells update their states synchronously with time based on a local rule. The neighborhood of the cell can be defined as the local set of positions that are connected to it at no more than a specified radius. In a dynamic CA, the state of the cell 'i' changes according to the states of its neighboring cells in a specific radius and itself. If the majority of the cell states surrounding it are 1's then it transforms its state to 1. The same applies if the majority of the cell states are 0's. This transition of states of the cells results in a dynamically evolving cellular automata. Figure 1 shows different types of neighborhoods for CA.

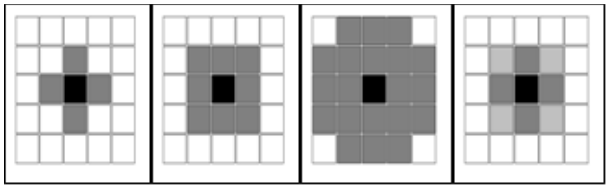

Fig 1: Different neighborhoods for Cellular Automata

2. LITERATURE SURVEY AND LIMITATION OF BASIC CA MODEL
The most popular CA model is most probably John Conway's Game of Life (GOL). Cells which can be either dead or alive are placed on a two-dimensional grid employing a Moore neighborhood. In spite of the simplicity of these rules, surprisingly complex patterns can emerge from special configurations. Still-life here refers to any configuration that remains stable under the GOL-rules. But there are also structures that oscillate, but reform on a different place – thus moving into a fixed direction. These structures are calls gliders [1]. Gliders are an example of a mobile pattern that after some generations completely replicates itself with an offset of some cells in vertical and/or horizontal direction. Glider can be constructed to move in any direction by simply rotating and/or reversing the arrangement of cells. Glider guns are able to generate gliders periodically. In figure 2, simple glitter is illustrated.

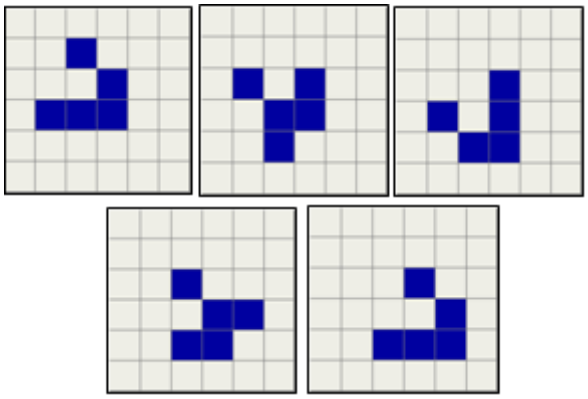

Fig 2: Simple glitter

3. PROPOSED SOLUTION

As we have seen, static rule model leads to the problem of glider. To overcome this problem, we have proposed a five stage solution for disease spreading problem. Table 1 describes the five stages of our model.

Table 1. Stages of model

Stage	1	2	3	4	5
Level of Infection	Not Infected	Initial Infected	Moderate Infected	Strongly Infected	Fully Infected

Each of the above stage has probability distribution and associated probability range that describes the severity level of disease. Stage 1 shows 0% infection of disease to a person and stage 5 shows 100% infected person. For example a person with probability value 1 is fully infected and our model assigns stage 5 to that person.

When a non–infected person (stage 1) interacts with one or more infected person then there is a possibility of spreading infection to non-infected person. In our model when a lower stage person (less infected) interacts with higher stage person (highly infected) the probability of infection/infection-level will increase. Figure 3 shows the state transition in our model.

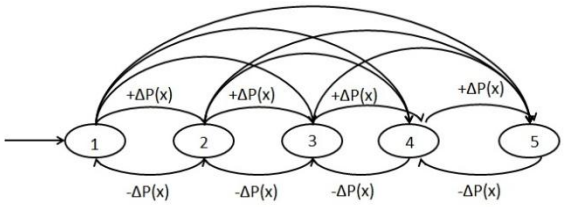

Fig 3: State transition diagram

$\Delta P(x) = \pm y$ (where y is dependent on probability of Moore neighbor's)

$T(m, n) = f(\Delta P(x))$ (Where T (m, n) denotes the change from state m to state n in state diagram)

NetLogo:

NetLogo [6] is particularly well suited for modeling complex systems developing over time. Modelers can give instructions to hundreds or thousands of "agents" all operating independently. This makes it possible to explore the connection between the micro-level behavior of individuals and the macro-level patterns that emerge from their interaction. Table 2 describes the different stages created for our proposed model in NetLogo and table 3 shows the various state change rules.

Table 2. Stages for NetLogo

Stage	1	2	3	4	5
Description	Not Infected	Initial Infected	Moderate Infected	Strongly Infected	Fully Infected
Probability Range	0	$0 < p \leq 0.3$	$0.3 < p \leq 0.5$	$0.5 < p \leq 0.8$	$0.8 < p \leq 1.0$
Color	Green	Blue	Yellow	Orange	Red

Table 3. State change rules

Current Stage	$P_{New} = P_{Old} + \Delta P$	Next Stage
1	$0 < p \leq 0.3$	2
1	$0.3 < p \leq 0.5$	3
1	$0.5 < p \leq 0.8$	4
1	$0.8 < p \leq 1.0$	5
2	$0.3 < p \leq 0.5$	3
2	$0.5 < p \leq 0.8$	4
2	$0.8 < p \leq 1.0$	5
3	$0.5 < p \leq 0.8$	4
3	$0.8 < p \leq 1.0$	5
4	$0.8 < p \leq 1.0$	5

Cell Update Algorithm:

Initialization Phase:

Create a user specified non-infected turtles in the NetLogo world with the following initial parameters:

1. Initial probability : 0 [Stage 1]
2. Assign stage as 1 (Color Green)
3. Put into a random patch in the NetLogo world.

Create a user specified infected turtles in the NetLogo world with the following initial parameters:

1. Initial probability : 1 [Stage 5]
2. Assign stage as 1 (Color Red)
3. Put into a random patch in the NetLogo world.

Infection and recovery phase:

1. Select a random turtle
2. Deduct the `natural deaths' from the turtle population (Optional).
3. Deduct the deaths due to continue infections (Optional).
4. Add to the population any newborns.(Optional)
5. Compute Moore's neighbor cell infection.
6. Compute infection probability.
7. Change the stage according to the probability.
8. Compute turtle recoveries.(Optional/if any remedies specified)
9. Simultaneously calculate the above steps for all turtles in the NetLogo world.

Movement phase:

1. Select a random turtle
2. Check the probability and change the color of turtle
3. Move turtle in random direction

4. SIMULATION RESULTS

For simulation, we have used NetLogo software. In NetLogo, we have created various scenarios & got the correct results. One of the scenarios is described below. In this scenario, the world has non-infected turtles 118 and infected turtle 84.

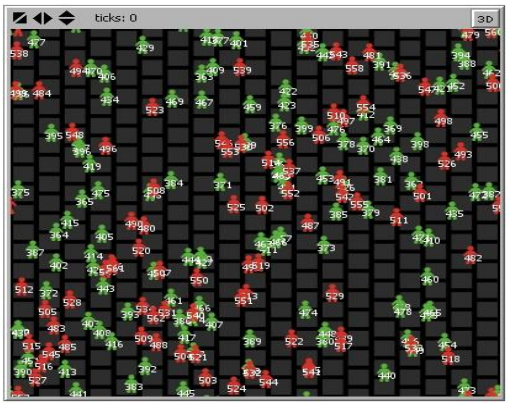

Fig 4: Initial setup

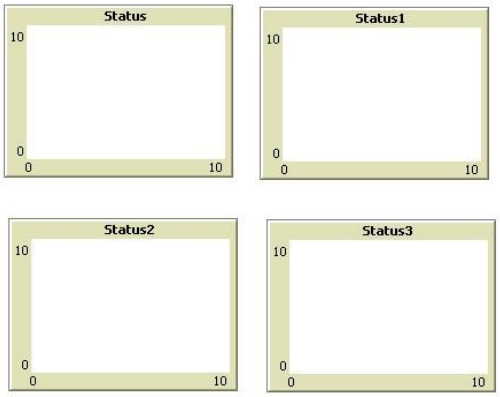

Fig 5: Status of initial setup

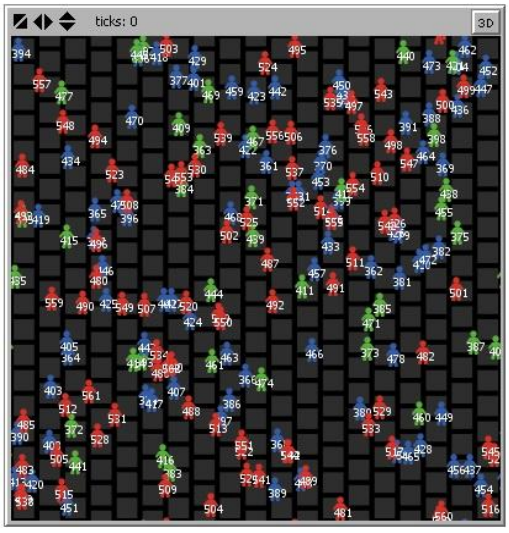

Fig 6: Ticks = 60

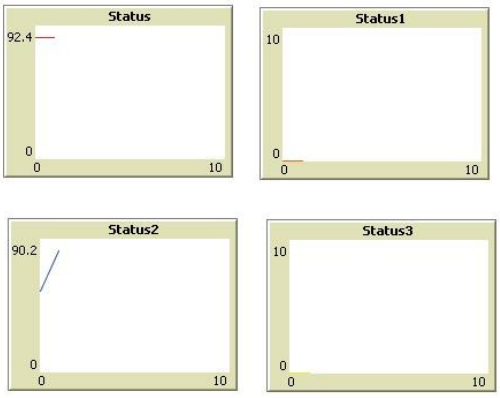

Fig 7: Status at ticks = 60

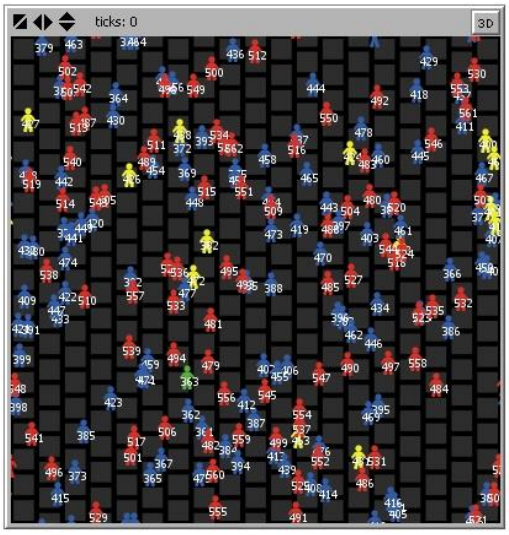

Fig 8: Ticks = 100

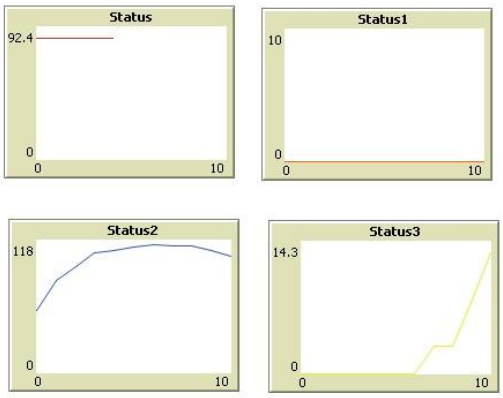

Fig 9: Status at ticks = 100

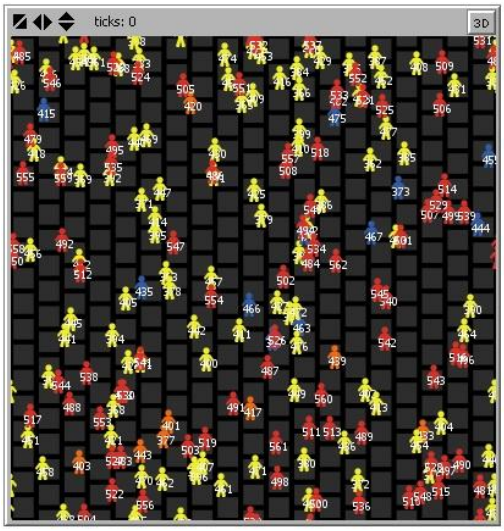

Fig 10: Ticks = 150

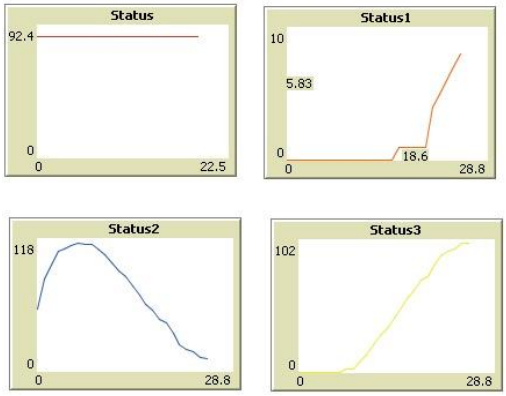

Fig 11: Status at ticks = 150

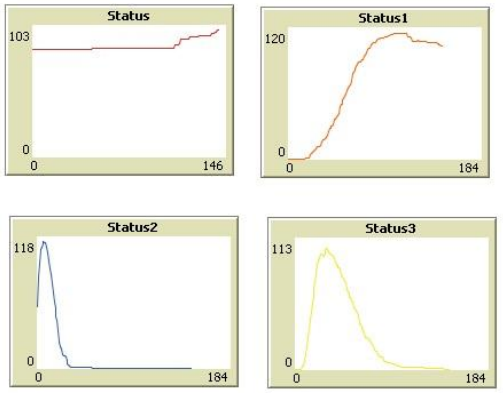

Fig 12: Status at ticks = 250

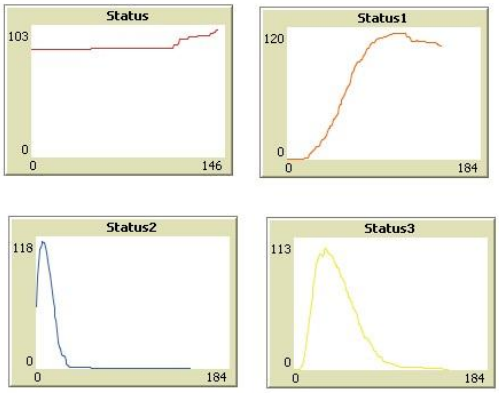

Fig 13: Status at ticks = 330

Figure 4 to 13 shows different stages of spreading of disease in a real life landscape with imaginary town centers and transport links. From the above status graph, we observe that in our model, the turtles are random with their probabilities and disease stages.

5. CONCLUSION

The proposed epidemic model provides an opportunity to demonstrate the capabilities of graphical CA model. In this work, we have shown the applicability of probabilistic model as a predictive tool for epidemiologist. Our model is useful for future planning and remedies of disease spreading. We conclude that the multi stage probabilistic model removes glider pattern from the basic CA model.

6. REFERENCES

[1] E. Sapin, L. Bull, "The Emergence of Glider Guns in Cellular Automata found by Evolutionary Algorithms", unpublished.

[2] Shih Ching F, George Milne, "Epidemic Modelling Using Cellular Automata", Proceedings of the First Australian Conference on Artificial Life 2003

[3] Bhavana N. Umrikar, Manisha B. Patil, Chitra G. Desai, "Application of cellular automata technique for prediction of growth pattern through Java programming", International Journal Of Geomatics And Geosciences, Volume 2, No 1, 2011

[4] Niloy Ganguly, Biplab K Sikdar, Andreas Deutsch, Georey Canright, P Pal Chaudhuri, "A Survey on Cellular Automata", Technical report, Centre for High Performance Computing, Dresden University of Technology, December 2003.

[5] Ali Yarahmadi, Nazanin Moarefi, Saeed Setayeshi, "Implementing Cellular Automata with Dissimilar Rule on Serial Base", IEEE Conference Publications

[6] "NetLogo Home Page", [Online]. Available: ccl.northwestern.edu/netlogo/docs