The Spatial Prisoner's Dilemma CA and Cell-DEVS Simulation using RISE Remote Simulation (Lopez Model)

Chidozie V. Analikwu Gabriel Wainer

Department of Systems and Computer Engineering
Carleton University
1125 Colonel by Drive
Ottawa, ON. K1S-5B6 Canada
{chidozieanalikwu, gwainer}@sce.carleton.ca

Abstract Social behavior has been under studies for decades with lots of questions been ask on why certain rational individuals will choose to exploit the weak for material gains or why nations will choose to go to war and overthrow other nations for economic benefits; with many more of this questions still left unanswered. This behavioral traits are not only observable in humans but a study of animal behavior show some degree of corporation but majorly characterized with survival (fight) of the fittest to ensure subsequent evolution of the genes of the stronger as described in Darwin's Theory of Evolution [9]. Experimental studies of this interesting phenomena will be overwhelming using live scenarios as so much time and effort will be expanded. This term paper builds upon the Spatial Cellular Automata (CA) model of the game "Prisoner's Dilemma" which has been canonized in game theory as an acceptable model for studying social interaction by extending this model to Cell-DEVS simulation environment using RISE (RESTful Interoperability Simulation Environment) remote simulation.

I. INTRODUCTION

The 21st century has witnessed tremendous innovation in the area of research and exploration as we have continue to see an ever increasing demand for supercomputers with inbuilt software simulation and analysis tools to ensure ease of computation, and accuracy in research result obtained. We are currently witnessing a paradigm shift to the age of cloud computing which has tremendous prospect in not only delivering the expected results but also cutting overall cost, improved collaboration between colleagues, better performance, scalability, flexibility, data recovery and backup, and lots more. We explore this advantages of cloud computing to our study of the game Prisoner's Dilemma using Cell-DEVS Simulation running on cloud computing technology RISE (RESTful Interoperability Simulation Environment). RISE is a simulation middleware to support RESTful-CD++ web services for remote simulation, which aims to support interoperability and mash-ups of distributed simulations regardless of the model formalisms, model languages or simulation engines [12].

There are so many question been ask that are yet to be answered such as why a rational individual will choose to cooperate at a certain time and at another time choose not. Prisoner's Dilemma is a game widely accepted in game theory which have provided a mechanism which we can easily employ to formulate some of this question and then use mathematical methods to analyze and draw some conclusions. In this paper we employ the Cell-DEVS software simulation to carryout simultaneous simulations of the Cellular Automata (CA) model of this game under different input parameters and startup configurations to draw a conclusions on this observable spatial changes as we relate this to social behavior.

II. BACKGROUND

Prisoner's Dilemma was originally framed by Merrill Flood and Melvin Dresher while at work at the

American Research and Development Corporation (RAND) in 1950. Albert W. Tucker formalized the game with prison sentence rewards and gave it the name "Prisoner's Dilemma". Tucker’s formalism follows that two criminal who are serving a year prison sentence are later discovered to be involved in more severe crime than the ones they are been punished for and the police not willing to let them get away with it seeks to find a way to pin the crime on them. But without sufficient evidence they resort to convincing either of the criminals into betraying the other. They propose the following to both of them

- If either of the prisoners betray the other while the other keeps silent the defector (termed the betrayer here) will be set free, while the cooperator (the one who keeps silent) gets additional 5 years jail time.
- If both of the prisoners were betray each other they both get two years jail time
- If both of them where to remain silent they will serve their remaining jail time of a single year.

	Prisoner B stays silent (cooperates)	Prisoner B betrays (defects)
Prisoner A stays silent (cooperates)	Prisoner A betrays (defects)	Prisoner A: 5 years Prisoner B: goes free
Prisoner A betrays (defects)	Prisoner A: goes free Prisoner B: 3 years	Each serves 2 years

Table I. Payoff Matrix of both prisoners in Tucker’s Prisoner’s Dilemma Formalism.

Assuming that the decision made by the prisoners have no effect on their future reputation, a careful look at the summary of the outcomes of the prisoner’s decisions otherwise known as the Payoff Matrix displayed in Table I gives the intuition that the best move for either of the prisoners will be to go with defection as this gives the maximum benefit and less risk of uncertainty when considering the other prisoners decision. We see a choice of betrayal more appealing than the benefit of Mutual Corporation which will better both parties. Several analysis of the prisoners decision can be made for instance if the prisoners where to play the game multiple times, the previous decisions in the other games will affect their present decision (termed Tit for Tat) and several more different scenarios like this can be presented.

Recently, it has been suggested by Nowak and May that spatial effect alone are sufficient to cause corporative behavior [8]. In their Cellular Automata (CA) model of the Prisoner’s Dilemma all of the sites of a two-dimensional lattice are occupied by players. The players interact with their nearest neighbor players in a pair-wise manner, over a number of time steps. The interaction strategies used by each player is determined as follows:

1. In a given time step, each player interacts with itself and with its eight nearest neighbors (e.g. the nine sites in the Moore neighborhood of the player if Moore Neighborhood is chosen) using either a cooperative strategy or an uncooperative strategy.
2. The total payoff to each player resulting from the nine interactions is determined, and each player adapts for the next time step the strategy of the player in its neighborhood (including itself) who received the biggest payoff.

		OPPONENT’S STRATEGY	
PLAYER’S STRATEGY		CCOOPERATE	DEFECT
	CCOOPERATE	1	0
	DEFECT	b	0

Table II. Payoff Matrix of a prisoner in Novak and May Prisoner’s Dilemma CA Formalism.

This simple and purely deterministic, spatial version of the Prisoner's Dilemma, with no memories among players and no strategically elaboration, can generate chaotically changing spatial patterns in which cooperators and defectors both persist indefinitely (in fluctuating proportions about predictable long-term averages). If the starting configurations are sufficiently symmetrical, these ever-changing sequences of spatial patterns-dynamic fractals-can be extraordinarily and beautiful, and have mathematical properties. There are potential implications of the dynamics to a wide variety of spatial extended systems in physics, politics, biology, law etc. [7].

We observe from Table II that the payoff matrix in Novak and May's formalism have been greatly simplified, using only '0', '1' and a value 'b' the reward for defection. In this paper we extend this framework of the Prisoner's Dilemma using Cell-DEVS simulation to analyze the effect of different values of the 'b' parameter rightly pointed out by Novak and May. Also we look at different startup (initial) configuration of the cells and try out a different neighborhood (with Von Neumann neighborhood in mind) in the aim to show that this spatial effect can also be observed with a different neighborhood and highlight the characteristic observed with this choice of neighborhood.

III. Defining the Model

Cell-DEVS which is based on CD++ specification language has been successful used to model and simulate complex cellular automata models having the advantage of evaluating the cells asynchronously with different timing delays. It provides an easy to use software visualization environment to study the behavior of these models which are specified using simple defined rules. This rules replicate the characteristic behavior of the system in real life and the output result based on this rules provides sufficient data and information to aid in our analysis. We will employ the use of a Lopez Model which is an extension of Cell-DEVS that permits cells to use multiple state variables and multiple ports for inter-cell communications to define our model for the Prisoner's Dilemma.

A. Formal Specification

The formal specifications $<X, Y, I, S, \theta, N, d, \tau, \delta int, \delta ext, \lambda, ta>$ for the atomic Lopez model of the Prisoner's Dilemma CA is defined as follows:

X = {∅} // Input external event

Y = {∅} // Output external event

I = <45, 4, {P^x}, {P^y}> // Model's modular interface with the neighborhood size $\eta = 45$,
number of other ports $\mu = 4$,
[{P^x} {P^y}] = {state, stage, cState, totalPayoff}, Which are all internal ports used to send and receive values from cells within the neighborhood of a cell.
Where [state] = used to exchange information on the current state and previous state between each cell and its neighbor.
[stage] = keeps information of the current phase of each cell.
[cState] = used to exchange information of current state between each cell and its neighbor.
[totalPayoff] = used to exchange information of the total payoff value between each cell and its neighbor.

S = {[11], [12], [21], [22]} // Possible states for a given cell. We can also choose to write this as {[CC], [CD], [DC], [DD]} respectively
Where [11] = denotes a player cooperating now and cooperated in the previous step [Color – Blue]
[12] = denotes a player cooperating now and defected in the previous step [Color – Green]
[21] = denotes a player defecting now and cooperated in the previous step [Color – Yellow]
[22] = denotes a player defecting now and defected in the previous step [Color – Red]

θ= { (s, phase, σqueue, σ)} //cells state variable
where $s \in S$ //defined above
phase = {1,2} // which stage defined above keeps track off.
where [1] = Payoff should be calculated.
[2] = Strategy should be updated.
σqueue = {(vi, σi) / i $\in$ N, vi $\in$[0, infinity), σi $\in$ R0+]

and $\sigma \in$ R0+ //uses transport delay

N = {∅} // set of input events

d = 100ms //delay for all individual cells

τ = local computing function which will be discussed in section B.

δint://internal transition function that is defined by CD++ automatically

δext://external transition function that is defined by CD++ automatically

λ = {∅} is the output function

ta(passive) = INFINITY

ta(active) = d

B. Detail Description and Implementation

The Lopez Model of the Prisoner's Dilemma has same basic rules which we have already discussed in our background study. We will go straight into how this rules are implemented in Lopez and necessary setup required. At first an initial choice of the cells geometry is decided, in this case we will be using a 2D cell space of 45 x 45 with Wrapped around borders and transport delay of 100 time units as shown in Fig. 1. The next step is a choice of Neighborhood. We start our studies using the nine sites of the Moore's Neighborhood as we can see already defined in Fig. 1.

```
[top]
components : PrisonersDilemma

[PrisonersDilemma]
type : cell
width : 45
height : 45
delay : transport
defaultDelayTime : 100
border : wrapped
neighbors : PrisonersDilemma(-1,1) PrisonersDilemma(0,1)
neighbors : PrisonersDilemma(1,1) PrisonersDilemma(1,0)
neighbors : PrisonersDilemma(-1,0) PrisonersDilemma(0,0)
neighbors : PrisonersDilemma(-1,-1) PrisonersDilemma(0,-1)
neighbors : PrisonersDilemma(1,-1)
initialvalue : 0
initialrowvalue :  21    000000000000000000000011100000000000000000000
initialrowvalue :  22    000000000000000000000015100000000000000000000
initialrowvalue :  23    000000000000000000000011100000000000000000000
localtransition : PrisonersDilemma-rule
neighborports : state stage cState totalPayoff
```

Fig. 1. Initial Configuration for the Lopez model of Prisoner's Dilemma CA.

The initial values for each cells are defined using the "InitialRowValue' as rightly shown in the Fig. 1. We then applied a rule based on this initial configuration to initialize the value of the 'ports' (Lopez model allows for definition of multiple ports) to our desired configuration. In this case our desire configuration will be, stage = 1, totalPayoff = 0 and the state will be discussed next. Let's consider a case of one defector in center (whom we assumed also defected in the previous step [22] or [DD]) surrounded by 8 defectors (whom we assume cooperated in the previous step [21] or [DC]) who are then surrounded by cooperators (whom we also assume here cooperated in the previous step [11]). Fig 2. shows the rules implemented to configure the ports as we already discussed while Fig.3 gives a pictorial view of how this initial cell configuration should look after a successful initialization.

```
rule : { ~stage := 1; ~cState := 1; ~state := 11; ~totalPayoff := 0; }
       100 { (0,0)~state = 0 }
rule : { ~stage := 1; ~cState := 2; ~state := 21; ~totalPayoff := 0; }
       100 { (0,0)~state = 1 }
rule : { ~stage := 1; ~cState := 2; ~state := 22; ~totalPayoff := 0; }
       100 { (0,0)~state = 5 }
```

Fig 2. Rule defining the initialization of the cells.

11	11	11	11	11	11	11
11	11	11	11	11	11	11
11	11	21	21	21	11	11
11	11	21	22	21	11	11
11	11	21	21	21	11	11
11	11	11	11	11	11	11
11	11	11	11	11	11	11

Fig 3. Display of Cells after initialization

The game is played over a number of rounds (i.e. time steps), in each of which every prisoner (site) on the lattice interacts with itself and with the eight nearest neighbor sites of the Moore Neighborhood. The game is subdivided into two stages:

1) ***Stage One:*** Each prisoner interacts with the prisoners in its Neighborhood in a pair-wise manner. And for each interaction a payoff value is computed as follows:
   - If the player and its neighbor both cooperate, the player gets a point
     [Cooperate + Cooperate = 1]
   - If the player and its neighbor both defect, the player gets nothing.
     [Defect + Defect = 0]
   - If the player defects and its neighbor cooperates, the player gets b points (where b>1). In this case our b values is chosen to be 1.85. Other values of b will be chosen in later case.
     [Defect + Cooperate = b]
   - If the player cooperates and its neighbor defects, the player gets nothing.
     [Cooperate + Defect = 0]

A sample rule defining the interaction of a cell (0, 0) with one of its neighboring cell (1, 0) is shown in Fig. 4. and subsequently the total payoff calculated for all interactions in one time step from initial configuration is shown in Fig.5

```
rule : {  ~stage := 2; ~state := (0,0)~state; ~totalPayoff :=

          if  ((0,0)~cState = 1 and (1,0)~cState = 1, 1, 0) +
          if  ((0,0)~cState = 1 and (1,0)~cState = 2, 0, 0) +
          if  ((0,0)~cState = 2 and (1,0)~cState = 1, 1.85, 0) +
          if  ((0,0)~cState = 2 and (1,0)~cState = 2, 0, 0) +
```

Fig. 4. Rule defining the interaction of cell (0, 0) with (0, 1) and payoff computed with b = 1.85

8	8	8	8	8	8	8
8	7	6	5	6	7	8
8	6	9.2	5.6	9.2	6	8
8	5	5.6	0	5.6	5	8
8	6	9.5	5.6	9.2	6	8
8	7	6	5	6	7	8
8	8	8	8	8	8	8

Fig 5. Display of Payoff computation for one time step after initialization.

2) ***Stage Two:*** Each prisoner now compares its total payoff value with itself and all the other prisoners in its neighborhood (Moore Neighborhood in this case) looking for the prisoner that is the most successful in that round. The prisoner will then update strategy to the one used by its neighbor to achieve the highest score.

A sample rule defining the update of strategy assuming the cell (-1, 0) has total payoff value greater than that of the neighboring cells is shown in Fig.6**.** Also the pictorial view of two time step from initial configuration showing a successful update of strategy is displayed in Fig.7. We observe that the defectors have increased in number as agreed to the prisoner's dilemma game where each prisoner sorts after the highest gain (payoff in this case).

```
rule : { ~stage := 1; ~state := (((-1,0)~cState * 10) + (0,0)~cState);
         ~cState := (-1,0)~cState; ~totalPayoff := 0; }

              100 {  (-1,0)~totalPayoff >= (0,0)~totalPayoff and
                     (-1,0)~totalPayoff >= (1,0)~totalPayoff and
                     (-1,0)~totalPayoff >= (1,1)~totalPayoff and
                     (-1,0)~totalPayoff >= (0,1)~totalPayoff and
                     (-1,0)~totalPayoff >= (-1,1)~totalPayoff and
                     (-1,0)~totalPayoff >= (-1,-1)~totalPayoff and
                     (-1,0)~totalPayoff >= (0,-1)~totalPayoff and
                     (-1,0)~totalPayoff >= (1,-1)~totalPayoff and
                     (0,0)~stage = 2
                   }
```

Fig. 6. Rule defining the update of strategy assuming that cell (-1, 0) achieved the highest total payoff value.

11	11	11	11	11	11	11
11	21	21	21	21	21	11
11	21	22	22	22	21	11
11	21	22	22	22	21	11
11	21	22	22	22	21	11
11	21	21	21	21	21	11
11	11	11	11	11	11	11

Fig. 7. Display of successful update of strategy at second time step

IV. Simulation Result

All simulation where carried out using a 2D cell space of 45 by 45 with wrapped around border unless otherwise stated.

A. Using Moore Neighborhood with b = 1.85 and initial cell configuration of Fig. 3.

The simulation result showed a symmetric pattern of defectors gradually invading an entire cell space filled with cooperators which is coherent with the result also achieved by Novak and May. This interesting result is showed in Fig.8**.**

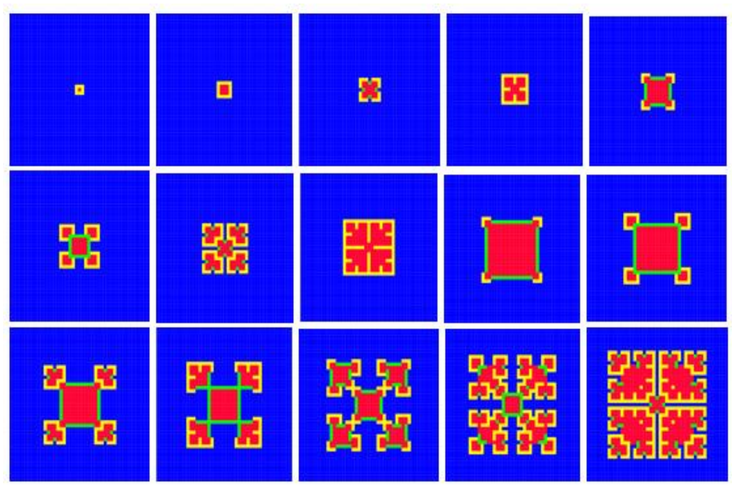

Fig 8. Simulation Result for b = 1.85 using Moore Neighborhood and startup configuration as shown in Fig. 3.

The graphical result of this simulation as shown in Fig. 9 shows a steep drop in the number of cooperators between time step [0 – 20] which was accompanied by increasing number of defectors. We notice that the frequency of cooperators gradually decays until the cooperators completely diminish. Fig. 9 show a graphical plot of the frequency of [CC]…[DD] against time steps while Fig. 10 shows an isolated case of only CC (labeled Cooperators) and DD (Labeled Defectors).

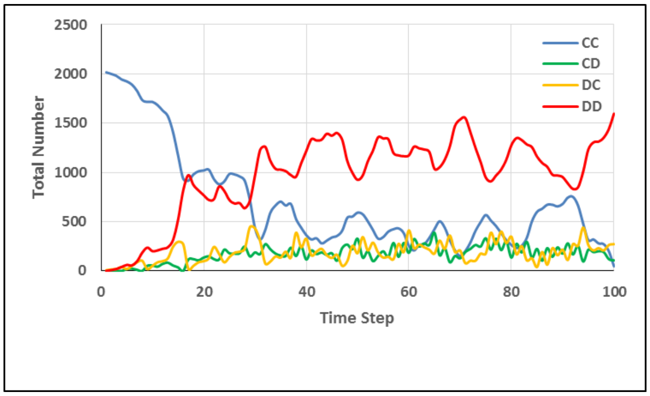

Fig. 9. A plot of the Total Number of [CC], [CD], [DC], [DD] against Time Step.

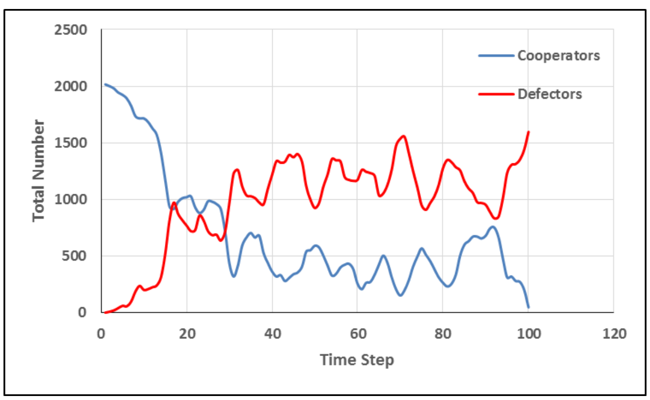

Fig. 10. A plot of the Total Number of Cooperators and Defectors against Time Step.

Something intrigued us as we viewed the simulation result with [b = 1.85]. We observed that areas that were at some point taken over by defectors where later at some other time step occupied later by cooperators. It was a struggle for dominance. Fig. 11 was extracted from the simulation result to buttress our observation.

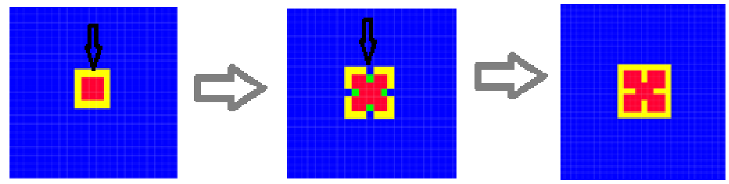

Fig. 11. Observation of cells previously occupied by defectors now taken over by cooperators.

The arrows in Fig. 11 are pointing to a cell that at time step = 4 had as its current value that of a defector i.e. [21] or [DC] (yellow) but in the next time step choose to be a cooperator [12] or [CD] (green). To better understand what is going on we extracted the payoff matrix of this cells before the transition as shown in Fig. 12.

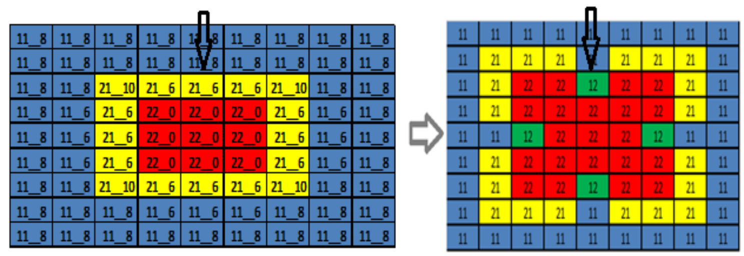

Fig. 12. Display of total payoff matrix and resultant update of strategy which resulted in cells previously occupied by defectors now taken over by cooperators. (Here for instance “21_6” denotes “state_totalpayoff”)

We observe here that due to the symmetry in patterns generated by the simulation result, this spatial arrangement at time step 4 favored the cooperators giving them more advantage around this cell in question as shown in Fig.12. Taking a closer look at the cell the arrow points to, the highest payoff recoded by the cells within the neighborhood of this cell is 8 which is that of a cooperator [CC] and that is why the state change to [12]. But in the next time step the symmetric pattern changes giving advantage to the defectors hence we observed that this same cell was later taken over by the defectors. This interesting phenomena is what leads to the alternating pattern observed as the Cooperators and Defectors battle for dominance of the 2D space. Since the benefit of defection is greater than that of corporation (i.e. b > 1) we eventually expect each prisoner or cell will go for the highest payoff which is that of defection.

B. *Using Moore Neighborhood with b = 1 and initial cell configuration of figure*

Simulating using b = 1 shows an interesting result, as we observed that having the payoff for defection equal to the gain of corporation, the defector at the center was not able to leave his neighborhood to occupy any cell instead his neighborhood was taken over by cooperators. The case where the price for betrayal is same with the price of corporation, any rational individual will choose to cooperate. The simulation result is shown in Fig. 13.

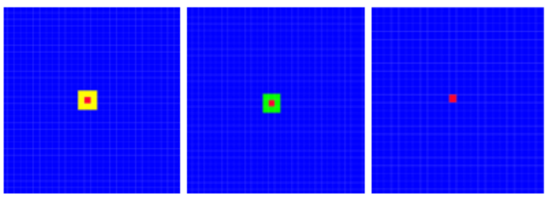

Fig 13. Simulation Result for b = 1 using Moore Neighborhood and startup configuration as shown in Fig. 3.

C. *Using Moore Neighborhood with b = 1.25 and initial cell configuration of figure*

Setting [b = 1.25] we observed an interesting alternating pattern of cooperators taking over the neighborhood of the defector in the center in one round and in the next round the defector takes over his neighborhood as shown in Fig. 14, a deviation from what was observed with b = 1. This same results was obtained for different values of b until we got to [b = 1.85]. The result at [b = 1.85] was maintained for [1.85 =< b < 2.05]. This alternating pattern was as result of the fact that the payoff for Mutual Corporation between cells that have the state of a cooperator far outweighs the benefits of mutual defection and hence we see the cells surrounded the defector acting like a boundary. In one time step the cooperators push towards the defector in the middle and in the next time step the defector push back.

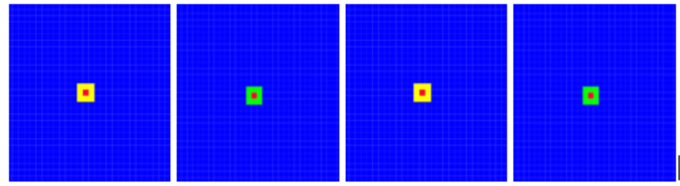

Fig 14. Simulation Result for 1.25 =< b < 1.85 using Moore Neighborhood and startup configuration as shown in Fig. 3.

D. *Using Moore Neighborhood with b = 2.05 and initial cell configuration of figure*

Using [b = 2.05] and above showed a rather interesting pattern as we saw the defectors forcefully making their way through the diagonals. This is due to the fact that the total payoff value for defection at the corners are very high and hence the defectors will want to take all they can get moving in that direction. Fig. 15 shows the simulation result obtained for this configuration.

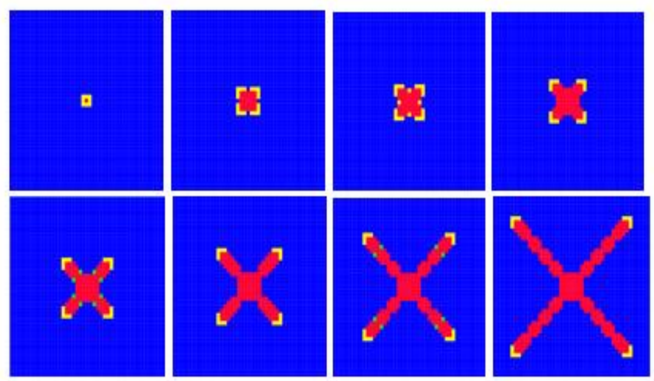

Fig 15. Simulation Result for b > 2.05 using Moore Neighborhood and startup configuration as shown in Fig. 3.

E. *Using Moore Neighborhood with b = 1.85 and random initial cell configuration*

So far we have explored several scenarios with the cells initialized as shown in Fig. 3. This characteristic of this configuration resulted in a symmetric pattern which was clearly observed (with b = 1.85 and 2.05). We present a case here where the cells are initialized with a random value for each cell. We observe from the simulation result obtained that using this random cell initialization and [b = 1.85], there was a loss in symmetric pattern we have seen so far as we expected. This simulation also showed that the rate at which the defectors take over the 2D space greatly depends on how many defectors at startup and how closely packed they are. Fig. 16 shows our simulation result with a random initialization of the cells.

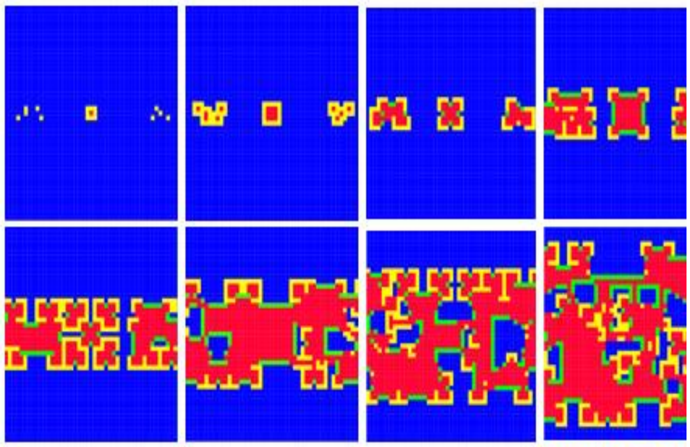

Fig 16. Simulation Result for b = 1.85 using Moore Neighborhood and random initial value for each cell at startup.

This same configuration was repeated for different values of b (i.e. b < 1.85 and b > 1.85). The simulation result for b < 1.85 was similar to that obtained in Experiment Frame III C. We observed just like in Experiment Frame III C the defectors not being able to push out of their boundary as a result of the benefit of mutual cooperation as we already explained. While for [b > 1.85] we saw an intriguing phenomena. The defectors which were initially defined in three separate units were able to form a bond within themselves and push towards the vertices as we previously observed in experiment frame III D. It was as if there exist an imaginary magnet or if to say an attraction that pulled the units together. The units which were at first separated at initial point now have a link or bridge between themselves. Fig. 17 shows a capture of this fascinating result.

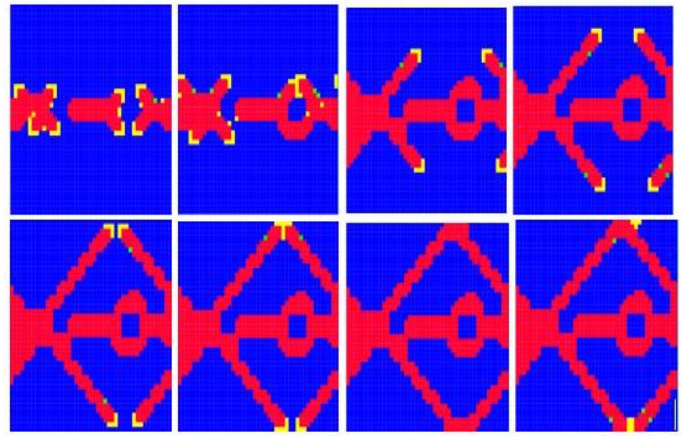

Fig 17. Simulation Result for b = 2.05 using Moore Neighborhood and random initial value for each cell at startup.

F. Using Von Neumann Neighborhood with b = 1.85 and initial cell configuration of Fig 18.

We conclude our simulation by showing that this spatial effect we so far studied using Moore Neighborhood is also achievable with other choice of Neighborhood. We show a final case study using Von Neumann Neighborhood and a startup configuration having one defector at the center (whom we assumed also defected in the previous step [22] or [DD]) surrounded by 4 defectors (whom we assume cooperated in the previous step [21] or [DC]) who are then surrounded by cooperators (whom we also assume here cooperated in the previous step [11] or [CC]). This initial configuration is shown in Fig. 18.

11	11	11	11	11
11	11	21	11	11
11	21	22	21	11
11	11	21	11	11
11	11	11	11	11

Fig 18. Display of Cells initialized for the case scenario using Von Neumann Neighborhood.

The result obtained using Von Neumann Neighborhood was very interesting as we observed a symmetric pattern as expected expanding from the center in the shape of a cross. The simulation result is shown in Fig. 19.

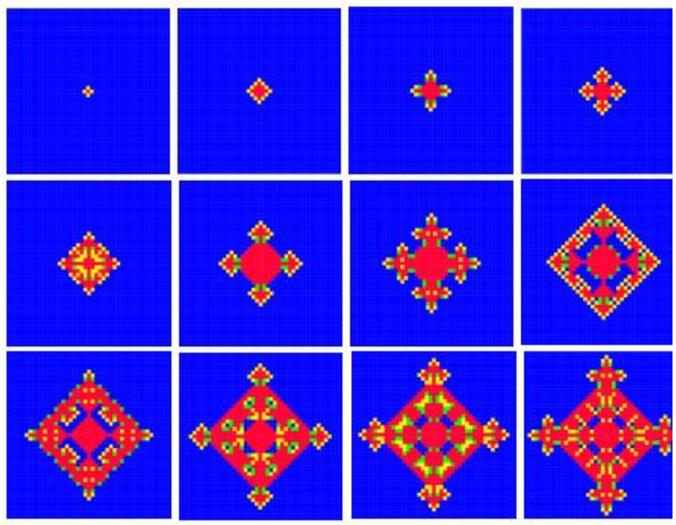

Fig 19. Simulation Result for b = 1.85 using Von Neumann Neighborhood and startup configuration as shown in Fig. 18.

A graphical analysis of this result is shown in Fig. 20 and Fig.21 while Fig. 20 shows the plot of all the states, Fig 20 isolated only the cooperators and defectors. We observe from Fig. 21 that between time step [0 – 30] there was a steep drop in the number of cooperators with an accompanying increase it the number of defectors. After time step 30 we see a maintained balance in the frequency of the cooperators and defectors.

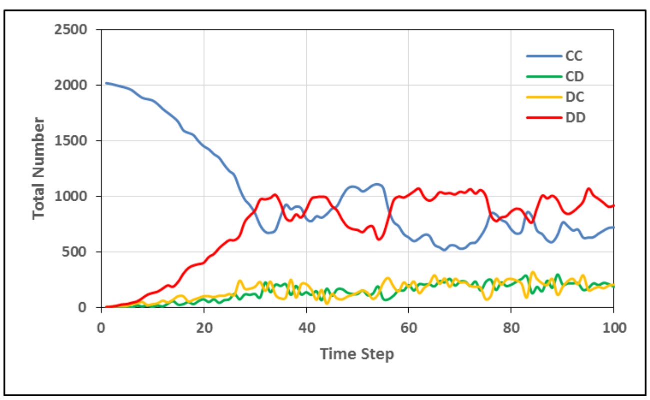

Fig. 20. A plot of the Total Number of [CC], [CD], [DC], [DD] against Time Step.

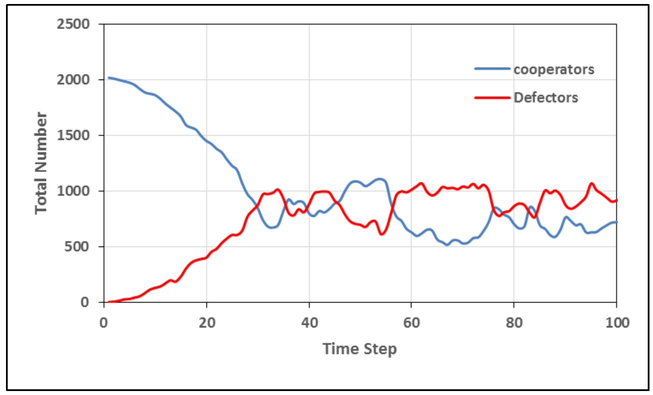

Fig. 21. A plot of the Total Number of Cooperators and Defectors against Time Step.

V. CONCLUSION

We presented how Cell-DEVs can be used to model and simulate the game Prisoner's Dilemma and how its unique features enabled us to have an elaborate visual display of our simulation result providing us with a better analysis and understanding of what is taking place at each time units. What we found most appealing is that running this extended version of Cell-DEVS (Lopez) on RISE greatly shortened the completion time for each simulation as compared to running this same model on the local machine. Therefore you need not worry about the limitations of your local machine or go through the strenuous task of installing a complex software to be able to visualize the results obtained in this paper. The RISE remote simulation web services has provided you with the needed comfort and convenience to take advantage of.

The simulation result presented here and analysis therein acts as a sample guide to interested individuals who in future intend using the CA model of the game Prisoner's Dilemma to model different scenarios in politics, biological sciences, social science, economics, law etc. We have showed that this model characteristics greatly depends on a careful choice of the parameter value for b, neighborhood style to use and the initial configuration of the cells. We have provided the needed framework for you to build on making it easy deciding which futures or characteristics as presented in the experiments above resembles your model and going ahead to start your implementation with a better understanding of how any changes in parameter will affect your model.

References

[1] G. Wainer, “Discrete-event Modeling and Simulation: a Practitioner's Approach,” CRC/Taylor & Francis, 2009.

[2] G. Wainer and N. Giambiasi: "Application of the Cell-DEVS Paradigm for Cell Spaces Modeling and Simulation", Simulation, Vol. 71, No. 1, pp. 22-39, January 2001.

[3] G. Wainer, "CD++: a Toolkit to Define Discrete-Event Models", Software, Practice and Experience, Wiley, Vol. 32, No 3. pp. 1261-1306. November 2002.

[4] A. López amd G. Wainer, “Improved Cell-DEVS model definition in CD++,” P.M.A. Sloot, B. Chopard, and A.G. Hoekstra (Eds.): ACRI 2004, LNCS 3305. Springer-Verlag. 2004.

[5] R. J. Gaylord and Kazume Nishidate, “Modelling Nature,” Springer-Verlag New York: TELOS, pp.174-183, 1996.

[6] A. L. Lloyd, “Computing bouts of the prisoner’s dilemma.” Scientific American, 272, 110-115, 1995.

[7] M. A. Nowak and R. M. May, “Evolutionary games and spatial chaos.” Nature, 359, 826-829, 1992.

[8] M. A. Nowak, R. M. May and K. Sigmund, “The arithmetics of mutual help.” Scientific American, 272, 76-81, 1995.

[9] C. Darwin, “The Origin of Species,” John Murray, Nov. 24, 1859.

[10] R. Axelrod, "Effective Choice in the Prisoner's Dilemma," Journal of Conflict Resolution 24: 3-25, 1980.

[11] R. Axelrod, “The Evolution of Cooperation,” New York: Basic Books, 1984.

[12] K. Al-Zoubi and G. Wainer, “Using REST Web Services Architecture for Distributed Simulation,” In: Proceedings of Principles of Advanced and Distributed Simulation (PADS 2009). pp. 114-121. Lake Placid, New York, USA, 2009.

[13] R. Axelrod and W. Hamilton, "The Evolution of Cooperation," Science 211: 1390-1396, 1981.

[14] E. Berlekamp, J. Conway and R. Guy, “Winning Ways for your Mathematical Plays III,” London: Academic Press, 1982.

[15] J. Demongeot, E. Golés, and M. Tchuente, “Dynamical Systems and Cellular Automata,” New York: Academic Press, 1985.

[16] A. Dewdney, “The cellular automata programs that create wireworld, rugworld and other diversions," Computer Recreations, Scientific American 262: 1, 146-149, 1990.

[17] G. Mar and P. St. Denis, "Chaos and Cooperation: Two-Dimensional Prisoner's Dilemmas in Continuous-Valued Logic," International Journal of Bifurcation and Chaos, August 1994.

[18] M. A. Nowak, “Stochastic Strategies in the Prisoner's Dilemma,” Theoretical Population Biology 38: 93-112, 1990

[19] M. A. Nowak, “The Spatial Dimensions of Evolution,” International Journal of Bifurcation and Chaos 3: 35-78, 1993.

[20] M. A. Nowak and K. Sigmund, “Game-Dynamical Aspects of the Prisoner's Dilemma,” Applied Mathematics and Computation 30: 191-21, 1989.

[21] M. A. Nowak and K. Sigmund, “Tit for tat in heterogeneous populations,” Nature 355: 250-252, 1992.

[22] M. A. Nowak and K. Sigmund, “A strategy for win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game,” Nature 364: 56-58, 1993.

[23] T. Toffoli and N. Margolus, “Cellular Automata Machines: A New Environment for Modelling,” Cambridge, MA: MIT Press, 1987.

[24] S. Wolfram, “Cellular automata as models of complexity," Nature 311: 419-424, 1984.

[25] S. Wolfram: “Theory and Applications of Cellular Automata Philadelphia: World Scientific, 1986.

[26] S. A. Frank, “Foundations of Social Evolution” Princeton Univ. Press, Princeton, Nov., 1998).