2D Simulation Of Stock Market Using Cell-DEVS

Yuhan Zeng, Gabriel A. Wainer
Dept. of Systems and Computer Engineering
Carleton University, Ottawa, Canada
6748530
ericzengyh@gmail.com

Abstract

Cell-DEVS (Cell Discrete Event System Specification) is a modular that is used for formalism for modeling and analyzing different natural and artificial systems. To study the behaviour in the real stock market, in this paper, we propose a model to simulate the dynamic of stock market using Cell-DEVS automata. In this model, we assume that there are two kinds of people, the active trader (represented by 1) and the negative people(represented by 0). The active trader will have a fixed influence on the negative people, that is , the more active traders around one cell, the higher probability it will become active. In addition, we also bring a state variable called changeability in every cell. If the changeability is 1, that means the cell cannot be changed by its neighbors. On the other side, if the changeability is 0, that means the cell can be changed by its neighbors. The simulation results and analysis for the variety states of stock market show that except those who are assumed to be unchanged, people in the stock market will be easily influenced by their neighbors and active traders will get together to form a large cluster, which is named herding behavior by socialist .

1. Introduction

In such a highly developed society, which can be characterized as a information exploding era, people have witnessed a trend that we are becoming more and more influenced by others. In many areas especially where the information plays an significant role such as media and stock market, people tend to blindly follow the others. This phenomena is defined herding behavior by socialists.

However, it is very difficult and costly to analyze this behavior in the real world. People are flowing, unexpected and private. And the information in the real stock market may be worth millions of dollars. Therefore, it is meaningful to use Cell-DEVS automata to model this phenomena.

One representative research about the complexity of stock market is done by Brian Arthur in Santa Fe Institute, New Mexico.

Researchers in Santa Fe Institute hold the opinion that, the economical system is related not only to the concrete things such as the scientists, many kinds of behaviors, markets, financial agencies, factories and so on, but also many psychological factors which hide behind them and have some interacts with them. It is these psychological factors that play important roles on macroeconomic behavior and promote the evolution of financial markets. [1]

Therefore, the investment psychology of the traders is the main factor of the stock market. This is also the starting point of our research.

In addition, the researchers in Santa Fe Institute found that the model of the stock market exits bubbles even the collapse of all the market. They also found that the bubbles or collapse is related to the traders' emotion such as greed, fear, and nervousness.

After that, there are also some researchers establishing many models about the stock market. In the phycological aspect of stock traders, some researchers point out that people in the stock market have many cognitive biases on the stock, including "uncertainty", "psychological", "loss aversion", "hindsight", "overconfidence", "excessive fear", "policy dependent psychology", "riches psychological", "gambling psychological", "herding mentality", "representative deviation", "emotional support", "anchor behaviors", "selection bias", "conservative bias ", and " framing effect".

These cognitive biases interaction in the market lead to excessive reflect of people's investment behavior. Thereby increase the shocks in the stock market. Those above cognitive biases are the bad consequence of the investment phycology.

Considering about the quantify and operability, in our model, the herding behavior is the only investment phycology we take into account. After determining the certain quantitative indicators, we try to figure out the potential relationship between the herding behavior and the investment phycology.

2. Background

Cell-Automata, which is also called also called cellular spaces, tessellation automata, homogeneous structures, cellular structures, tessellation structures, and iterative arrays, is a model that uses a simple coding and imitation cell propagation mechanism of the numerical algorithm spatial analysis. Taking the limited discrete state of each cells spreading in the Lattice Grid, following the same rules of action, updating according to certain local rules, a large number of cells make up of dynamic system evolution through the simple interaction.

There are two kinds of cell spaces: Von Neumann cell neighborhood(as figure 2.1 shows) and More cell neighborhood (as figure 2.2 shows).

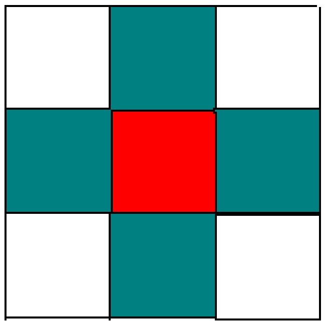


Figure 2.1 Von Neumann cell neighborhood

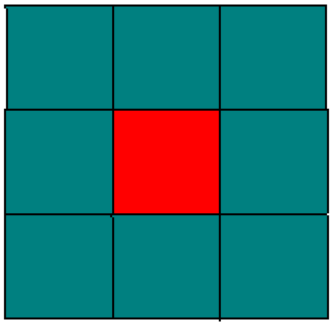


Figure 2.2 More cell neighborhood

From the above figure, we can know the neighborhood of the cell. For the Von Neumann cell neighborhood, assuming the central cell is (0,0) (as the red one shown in the figure), then its neighbors are (0,1), (0,-1), (1,0), (-1,0) (as the green one shown in the figure). Similarly, for the More cell neighborhood, assuming the central cell is (0,0) (as the red one shown in the figure), then its neighbors (as the green one shown in the figure) are (0,1), (0,-1), (1,0), (-1,0), (1,1), (-1,1), (-1,-1), (1,-1).

The description of cell space is reused from the contents in the class.

The specifications of cell space is as follows:

CCA = < S, n, C, h, N, T, t >

C: cell's state variables;
S: finite alphabet to represent each cell's state;
N: dimensional space;
N: neighboring cells;
T: global transition function;
t: local computing function;

In addition, the atomic Cell-DEVS is defined as follows(also reuse the resources in ppt)

CD = < X, Y, I, S, q, N, d, dint, dext, t, l, D >

X: the set of input external events;
Y: the set of output external events;
I: the model's modular interface;
N: the neighborhood size;
S: the set of possible states for a given cell;
q: definition of the cell's state variables;
N: the set of the input events;
d: the delay of the cells;
dint: the internal transition function;
dext: the external transition functiont
t: the local computing function;
l: the output function;
D: the duration function;

Also, the coupled Cell-DEVS is defined as follows(also reuse the resources in ppt)

GCCb = < Xlist, Ylist, I, X, Y, h, N, {r, c}, C, B, Z, select >

X: he set of external input events;
Y: the set of external output events;
D: an index of the components of the coupled model;
Md: a DEVS basic model
C: an atomic component;
B: if the cell space is wrapped;

Zdj: d to j translation function
Select: the tie-breaking selector;

There are other defines about the Cell-DEVS model which is also important, especially when programming.(still reuse the resources in ppt)

Neighborhood: coupling of model components;
Cell-space size: the size of cell space;
Borders (wrapped cell-spaces or self-state generating borders): the border character of cell space;
Initial state for the cell space: each cells ' initial state for cell space;
Priorities to treat simultaneous events: to classify the imminent cells in the cell space.
Array of atomic models and coupling: automatically created.

3. Model Defined

3.1.General Defined Model

In the stock market, investors are relatively independent to each other. At the same time, however, a strong behavior of dependence is commonly exited among the investors, which can be characterized as the "herding behavior". Extreme, if most of the investors have the same dependent behavior, intermittent events, like crashes or bubbles will happen. In my proposal, I will mainly describe the herding behavior of the stock market dynamics.

In the cellular automata model, cells on a 2D grid represent the agents of the market. And there are two states for each cell, which is σi(t) = 0 or 1, to represent the agents active (σi(t) =1) and inactive (σi(t) =0). Additionally, we also bring a state variable called changeability in every cell. If the changeability is 1, that means the cell cannot be changed by its neighbors. On the other side, if the changeability is 0, that means the cell can be changed by its neighbors.

In addition, we assume that the neighbors of influence are those of Von Neumann cell neighborhoods(shown in figure 3.1) .

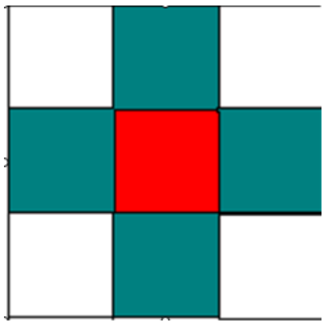


Figure 3.1 Von Neumann cell neighborhood

In the Von Neumann cell neighborhoods, we assume that the central cell is (0,0) (as the red one shown in the figure). And it has 4 neighborhoods(as the green one shown in the figure), which are: (0,1), (0,-1), (1,0), (-1,0). Each of the neighborhood cell has the same impact on the central cell. The more active neighborhood cells the central cell have, the larger probability it will become or remain active.

For each of the cell, some probabilities should be defined to have their own changes to the neighbors in order to simulate the people ' s characters in the real stock market. To simplify the model, we assume that there are only two kinds of people: active traders and passive potential traders with two kinds of characters: changeable and unchangeable. Therefore, there are three parameters to fix the percolation:

Pe: the probability of an active trader can change one of his inactive neighbors into an active one at the next time step (σi(t) = 0 → σi(t + 1) = ±1).

Pd: the probability that an active trader becomes passive due to his inactive neighbor (σi(t) = 1 → σi(t + 1) = 0). This simulates that when there is just one active trader, he/she will quit the market. In order to simplify the model and test different activities, I fix the value = 0.4.

Pa: the probability that an inactive cell change into an active one. This means that the non trading cell spontaneously decides to enter the market by themselves (σi(t) = 0 → σi(t + 1) = ±1). And I fix the value =0.01.

These three parameters will have a great effect on the stability of the stochastic trading system. In order to simplify the model, we assume that Pd and Pa are fixed. That is, the probability that an active trader becomes passive due to his inactive neighbor and the probability that an inactive cell change into an active one is unchangeable. In the real stock market, it means that if there is no active traders around, an active trader will become passive with a stable probability and even in an empty area without any active traders, some potential traders will also enter the stock market automatically.

By tuning different value of Pe, we can easily test different market activities.

However, the definition of Pe is the central cell's impact on its neighborhoods rather than the neighborhoods' impact on the central cell. In order to make it more close to our model and easier to simulate, we have to make a convert, using the formula below:

Pi: the probability that the central cell failed to influence its neighborhoods.

Pi= 1- Pe

P（n）: the impact of n active neighborhoods to the central cell.

P(n)= 1-Pi^n

Therefore, by using these two formular, we can convert the Pe into P(n). By tuning the value of P(n), we can analyze the impact of different amount of active neighbors bring to the central cell. The figure is as follows

	2	3	4	5
0.3	0.3	0.51	0.6572	0.7599
0.4	0.4	0.64	0.784	0.8704
0.45	0.45	0.6975	0.8336	0.9085
0.455	0.455	0.703	0.8381	0.9118
0.46	0.46	0.7084	0.8425	0.915
0.47	0.47	0.791	0.851	0.9211
0.485	0.485	0.7348	0.8634	0.93
0.5	0.5	0.75	0.875	0.9375

Figure 3.2 different probabilities

In addition, considering different characters of different people, we should bring in another state variable called changeability. To simplify the model, we just consider the two main situations, that is, there are only two state for each cell------changeable and unchangeable.

To the changeable cells, we can just follow the defining rules and see what is happening. However, to the unchangeable cells, the defining rules are useless.

This simulates the fact that, some negative people in an area, will never enter the stock market, which is quite common in our daily life. Also, even in the tough situation such as there is no active traders around, some stubborn active traders will also have a stable probability to remain active rather than quit the market. By using the state variable, we can get a better model which is closer to the real life.

The simulation steps is as follows:

At first, we load a small amount of active traders randomly and then evolve the previous rules to simulate. No matter what the Pe is, the initial value are the same as follows.

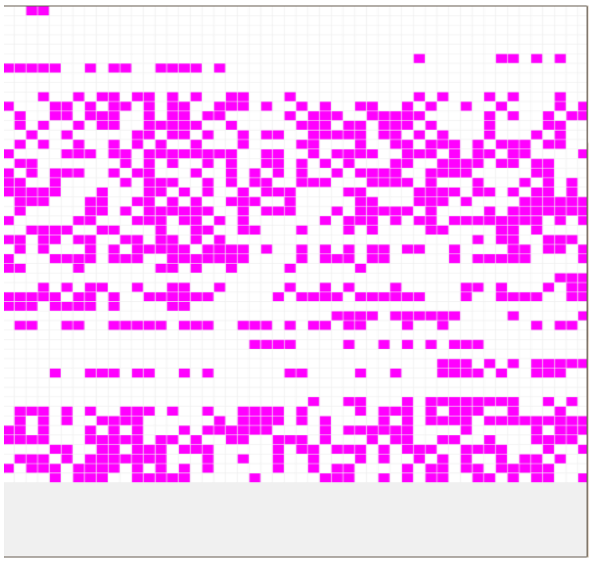


Figure 3.3 the initial value

The expected result is that the higher probability the Pe is, the more active traders there will finally be. That means, in the real market, the traders who have a larger number of sources of information, will have stronger impact on the others compared to the occasional investor.

Another expected result is that the active cells will get together to form a large cluster, just as mentioned before. This is called herding behavior.

Comparing with different values of state variable, we may also get a conclusion that the more changeable cells there are, the more clearly herding behavior there will be.

3.2 Formal Specification

The formal specification

<X, Y, I, S, N, d, δint, δext, λ, ta> is described as follow:

d=10 ms

ta (passive) = INFINITY

ta (active) = d

(will be written in the following descriptions

δint, δext, λ have been done by CD++.

3.4 Initial cell model description

The initial cell model is simple, idealized, and without any state variables. They can achieve the basic goals of the simulation.

There are two states of the cell: active (represented by 1) and inactive (represented by 0). All the cells will have initial values (1 or 0) which are randomly generated and will be changed by their neighbors with probability. Each active cell has its own stable probability to change its neighbors. That means, on the contrary, the more active neighbors around one cell, the larger probability it will be active. This mimics the fact that one people in an area with a lot of traders will easy to join in the stock market.

In addition, when there are no active neighbors around, an active cell will change into inactive with a probability.This means that in an area with no traders the active trader will possibly quit the market.

Also, in an empty area, without any active cells, there should exit a very little probability that an active cell will appear. This means that the non-trading cell spontaneously decides to enter the market by themselves.

All the cells will have a random value at the beginning of the simulation. Then they will several rules.

These rules can be divided into four parts with three parameters, and a default rule. In order to simplify the model and test different activities, I fix the value Pd= 0.4 and Pa=0.01 and change the value of Pe.

Because the value of and are fixed, we can easily define the following rules:

rule : 0 10 { truecount = 1 and round(uniform(0,10000))< 4000}
rule : 1 10 { truecount = 0 and round(uniform(0,10000))< 100}

These rules above show that if there is only one trader in the cells and it will easy to become inactive. Also if there are no traders at all, the inactive people will have little probability to become active traders.

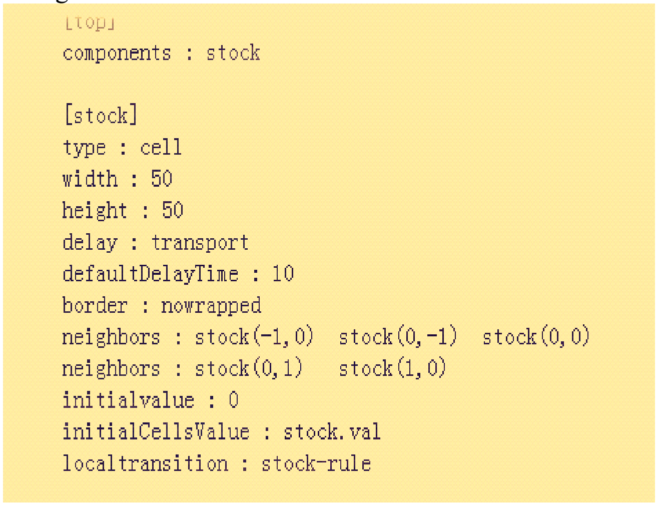
Then I will change the value of . Because Pe is the probability of one cell change the neighbors, we have to convert it into the changing probability by the neighbors. Thus, we use the function =1-P(n)^(n-1) (n represents the number of the active traders and n>1, because when n =1, the active trader will quit the market rather than influence others). The larger the n is, the higher probability there will be. The rules is as follows (Pe= 0.455)：

rule : 1 10 { truecount = 2 and round(uniform(0,10000))< 4550}
rule : 1 10 { truecount = 3 and round(uniform(0,10000))< 7030}
rule : 1 10 { truecount = 4 and round(uniform(0,10000))< 8381}
rule : 1 10 { truecount = 5 and round(uniform(0,10000))< 9118}

Also, the default rule shows the opposite condition, with the probability (1-P(n)) to change into inactive. The rule is as follows:

rule : 0 10 { t }

Before the rules, there will be some other definitions, which shows the parameters of simulation. As figure 3.4 shows below:



```
[top]
components : stock

[stock]
type : cell
width : 50
height : 50
delay : transport
defaultDelayTime : 10
border : nowrapped
neighbors : stock(-1,0)  stock(0,-1)  stock(0,0)
neighbors : stock(0,1)   stock(1,0)
initialvalue : 0
initialCellsValue : stock.val
localtransition : stock-rule
```

Figure 3.4 the definition of the rules

From the above rules, we can get that, the initial value is 0. And the stock.val offer the initial value of each cell. All the cell obey the stock rule which has been explained before.

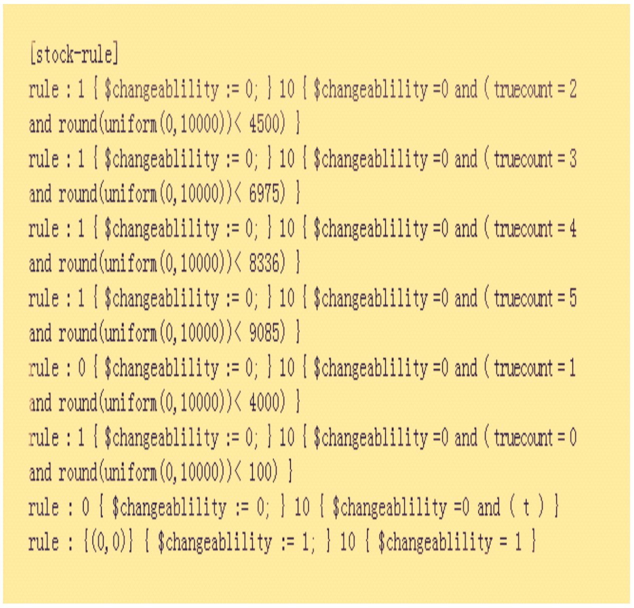
3.5 Advanced cell model description

In the initial Cell-DEVS model description, we assume that every cell are the same, which means that in the real stock market, everyone are in the same character. However, there is no doubt that this is too idealized. In fact, in the real stock market, different people will have different degrees of willing to do stock business. Some people will be easier to change while some are not. To simplify their different characteristics, we assume that there are only two kinds of characteristics of people in the stock market: changeable and unchangeable. To an active trader, if he or she is changeable, then the rules will be effective. However, if he or she is unchangeable, then the rules are useless. On the other hand, to a passive potential trader, same thing will also happen.

In order to make it in our model, we add a state variable called changeability. If the changeability equals 1, that means the cell is unchangeable. And if the changeability equals 0, that means the cell is changeable.

Then the old rules should be changed. New rules and gramma will be added.

These rules can be divided into five parts: three parameters, one state variable(value) , and a default rule. The definition of three parameters and the default rule are not changed too much. However, by adding a state variable, the code will be slightly different. The rules is shown in figure 3.5.

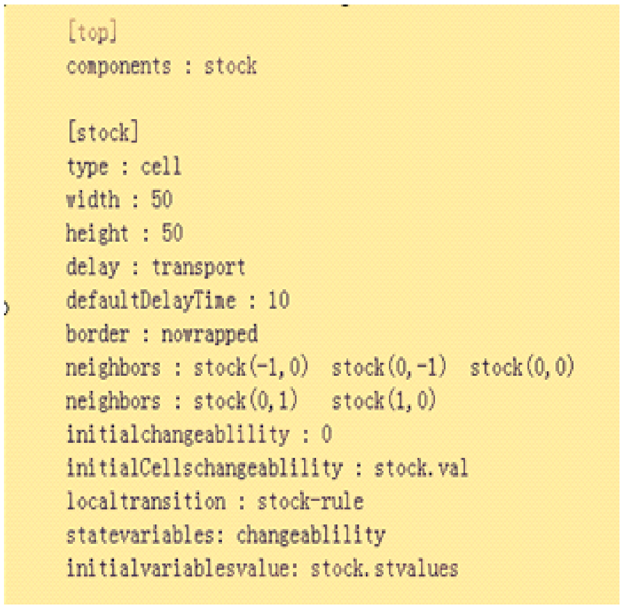


```
[stock-rule]
rule : 1 { $changeablility := 0; } 10 { $changeablility =0 and ( truecount = 2
and round(uniform(0,10000))< 4500) }
rule : 1 { $changeablility := 0; } 10 { $changeablility =0 and ( truecount = 3
and round(uniform(0,10000))< 6975) }
rule : 1 { $changeablility := 0; } 10 { $changeablility =0 and ( truecount = 4
and round(uniform(0,10000))< 8336) }
rule : 1 { $changeablility := 0; } 10 { $changeablility =0 and ( truecount = 5
and round(uniform(0,10000))< 9085) }
rule : 0 { $changeablility := 0; } 10 { $changeablility =0 and ( truecount = 1
and round(uniform(0,10000))< 4000) }
rule : 1 { $changeablility := 0; } 10 { $changeablility =0 and ( truecount = 0
and round(uniform(0,10000))< 100) }
rule : 0 { $changeablility := 0; } 10 { $changeablility =0 and ( t ) }
rule : {(0,0)} { $changeablility := 1; } 10 { $changeablility = 1 }
```

Figure 3.5 the rules of stock market

From the rules we can find that, there will be one more state variable called changeability, which represents the changeability discussed above. Before judging the number of active cells, we should judge the the cell’s state variable(changeability). If the changeability is 0, then the rules are effective. If the changeability is 1, then the rules cannot be used and the cell remain what it is before.

Before the rules, there will also be some other definitions, which shows the parameters of simulation. As figure 3.6 shows below:



```
[top]
components : stock

[stock]
type : cell
width : 50
height : 50
delay : transport
defaultDelayTime : 10
border : nowrapped
neighbors : stock(-1,0)  stock(0,-1)  stock(0,0)
neighbors : stock(0,1)   stock(1,0)
initialchangeablility : 0
initialCellschangeablility : stock.val
localtransition : stock-rule
statevariables: changeablility
initialvariablesvalue: stock.stvalues
```

Figure 3.6 the definition of the rules

From the figure above, we can figure out that there will be two more sentences. One is state variables, which define the name of the state variable. And the other is initial variables values, which define the initial value of each cell. Another difference is the attribute of the initial state values. It is not the val file but the stvalue file.

4. The Result

4.1 The Initial Cell-DEVS Simulation Result

By tuning the Pe we can get the figures as follows:

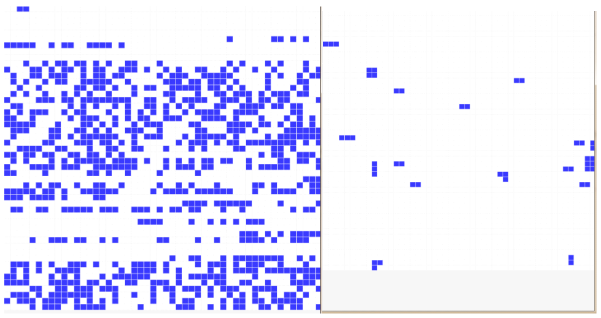


Figure 4.1 Pe=0.3

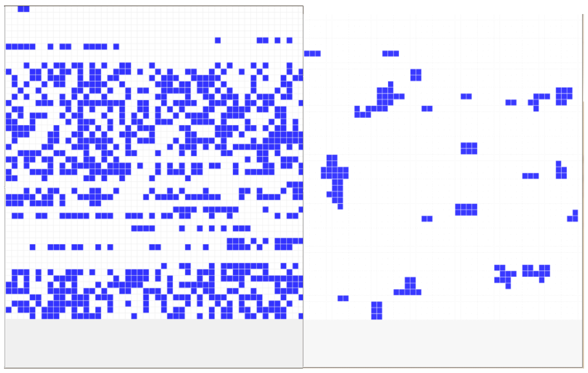


Figure 4.2 Pe=0.4

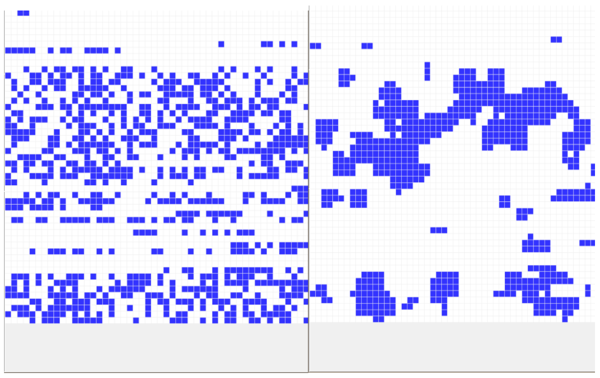


Figure 4.5 Pe=0.46

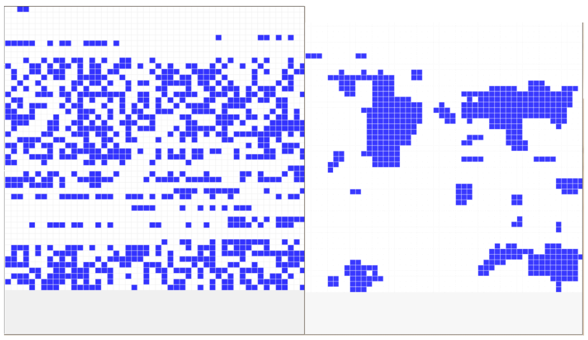


Figure 4.3 Pe=0.45

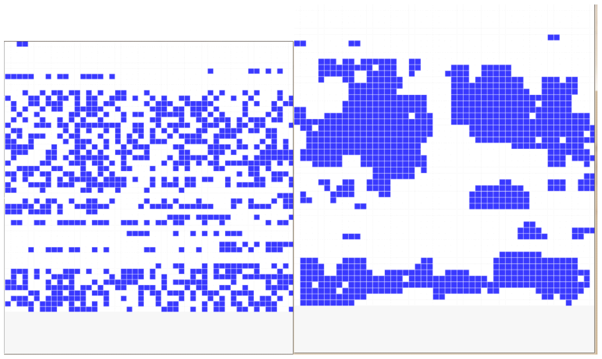


Figure 4.6 Pe=0.47

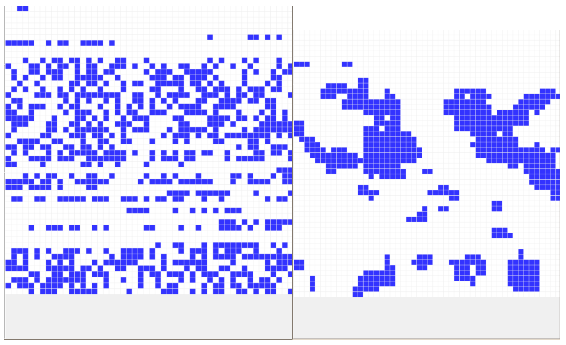


Figure 4.4 Pe=0.455

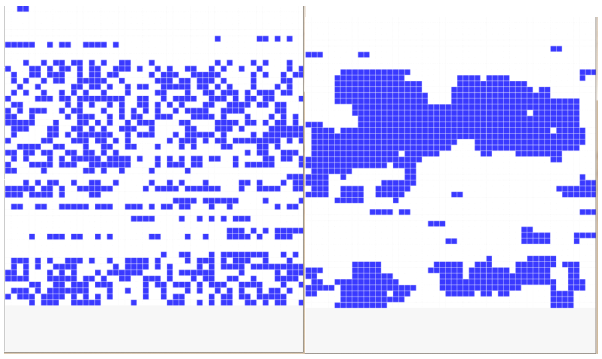


Figure 4.7 Pe=0.485

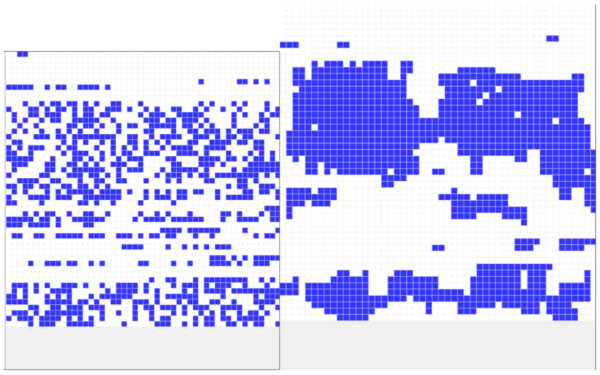


Figure 4.8 Pe=0.5

From above figures, we can easily recognize the herding behavior as we expected. The active traders gather together to become a large cluster. This means the traders in the stock market tend to be together which will help them do business with each other.

Also, we can recognize that the higher Pe is the more active traders there will be. That means, in the real market, the traders who have a larger number of sources of information, will have stronger impact on the others compared to the occasional investor.

4.2 Advanced Cell-DEVS Simulation Result

By adding a state variable, we can get the different results, as shown below(the initial state is just as the initial Cell-DEVS result). In order to make a better comparison, we put the initial Cell-DEVS result and the advanced Cell-DEVS result together.

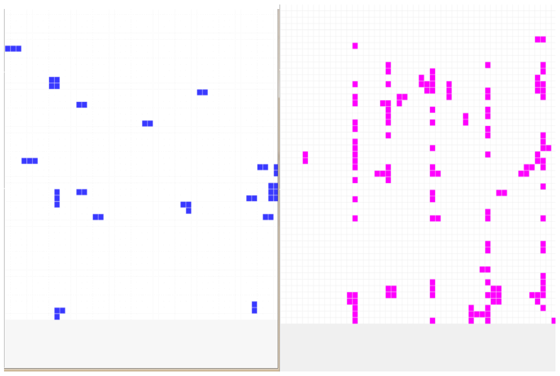


Figure4.9 Pe=0.3

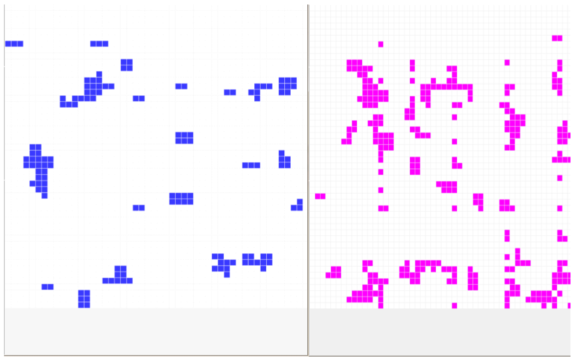


Figure4.10 Pe=0.4

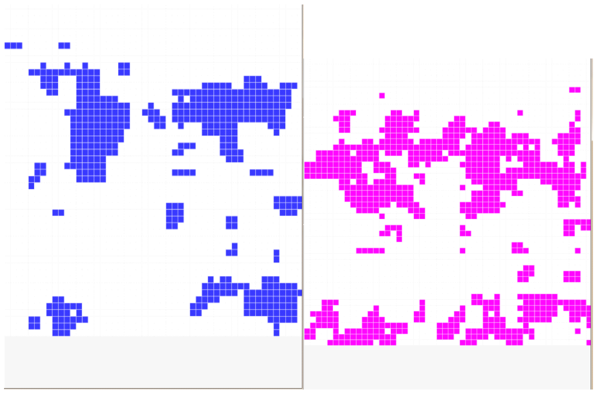


Figure4.11 Pe=0.45

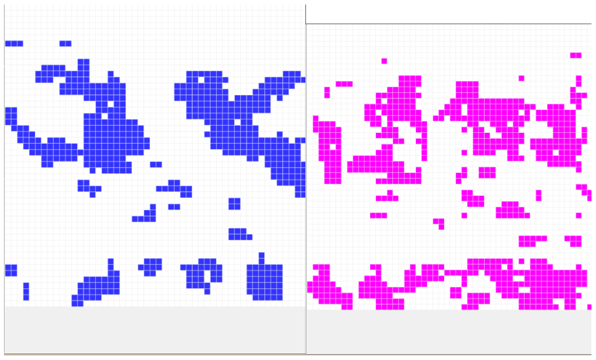


Figure4.12 Pe=0.455

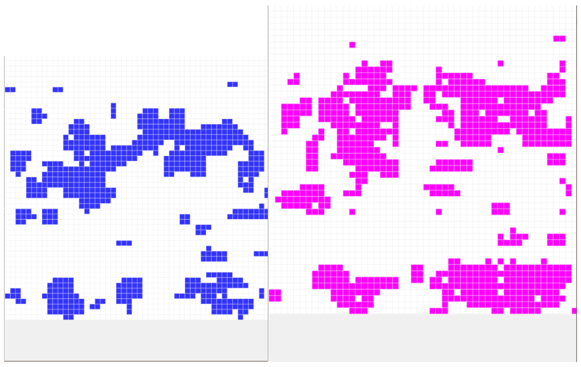


Figure4.13 Pe=0.46

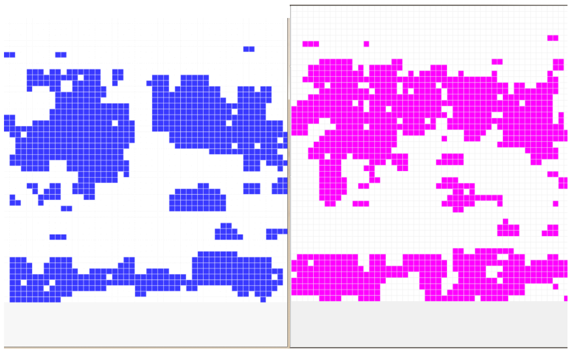


Figure4.14 Pe=0.47

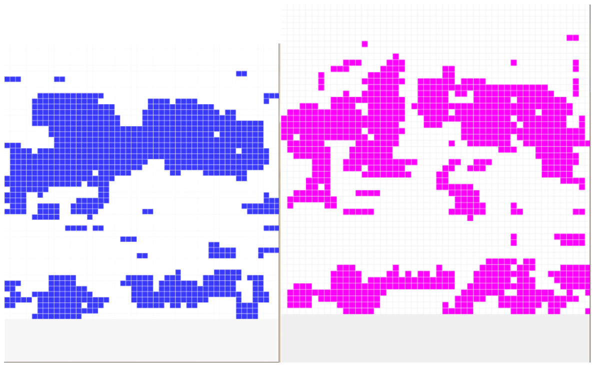


Figure4.15 Pe=0.485

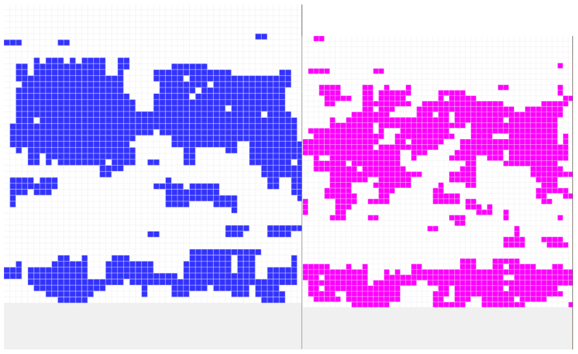


Figure4.16 Pe=0.5

From those figures above, we can find the differences when the state variable is considered.

First of all, it still obeys the rule that the larger Pe is, the more active cells there will be.

However, although we can still find out the phenomena of cluster, it is not so obvious comparing with the result of initial Cell-DEVS model. This is because that the state variable is brought in and some of the cells will not follow the defining rules.

Also, with the new state variables brought in, there appears to be another issue, that is, the distribution of the changeability. To test the influence of changeability, we should fix some of the rules to make the result more explicit.

Therefore, we change the previous rule:

```
rule :
{(0,0)} { $changeability := 1; } 10 { $changeability =
1 }
Into:
rule : 2 { $changeability := 1; } 10 { $changeability =1
and ((0,0) = 0) }
rule : 3 { $changeability := 1; } 10 { $changeability =1
and ((0,0) = 1) }
```

That means, when the initial value of the cell is 0 and the changeability of this cell is 1, then after some steps, the value of it becomes 2. Also, when the initial value of the cell is 1 and the changeability of this cell is 1, then after some steps, the value of it becomes 3.

The final results are as follows:

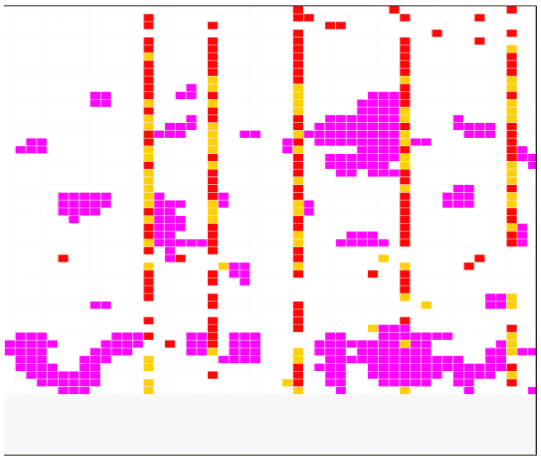


Figure 4.17 show the result of state value

From the picture, we can know that with the changeability of 1, the cell will remain its value and cannot be changed.

Because the output changeability will effect the visual effect, we just ignore it.

Assuming the number of cells whose changeability is C. When C is small, we can see the obvious cluster as mentioned above.

However, if we increase the value of C, then we can get the figures below:

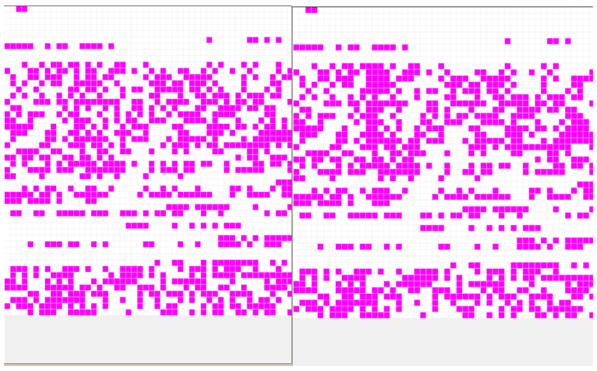


Figure 4.18 Pe=0.45 C=2285(total number is 50*50)

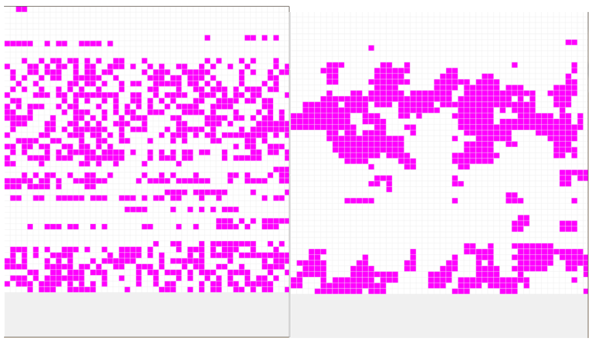


Figure 4.19 Pe=0.45 C=215(total number is 50*50)

Therefore, we can get the conclusion that, the higher the C is , the less obvious the cluster will be.

This phenomena is not very difficult to explain. Because the C represents the number of unchangeable cells. If it is big enough (such as 2285), then all of the cells are unchangeable. Thus, the rules are useless or can just be used in a small areas.

4.3 The Detail Explain Of the Initial Cell-DEVS

The detail explanation of the initial Cell-DEVS is as follows (take Pe=0.45 as an example):

Step 0: have the random initial value

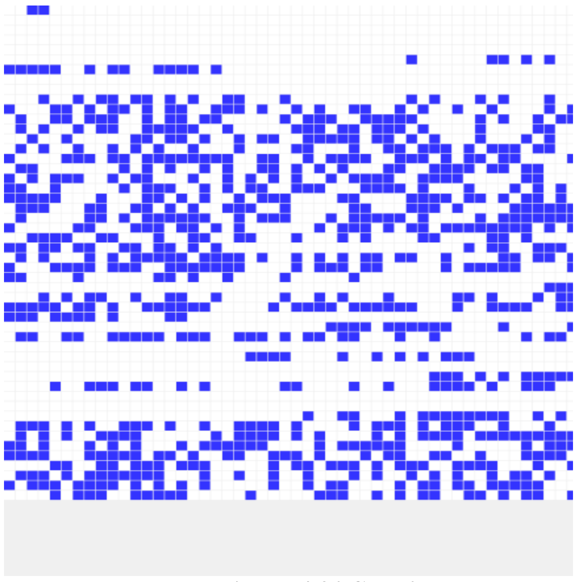


Figure 4.20 Step 0

Step 1: the first change:

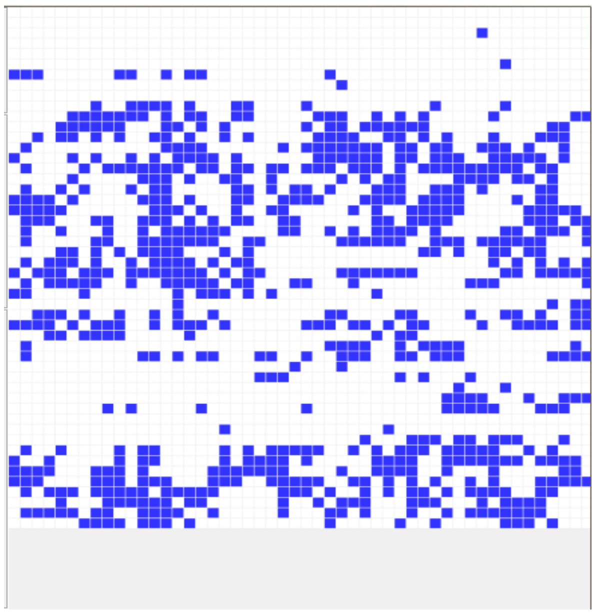


Figure 4.21 Step 1

With the first change, we can see some of the clusters get together. However, it is not so obvious.

Step 2:

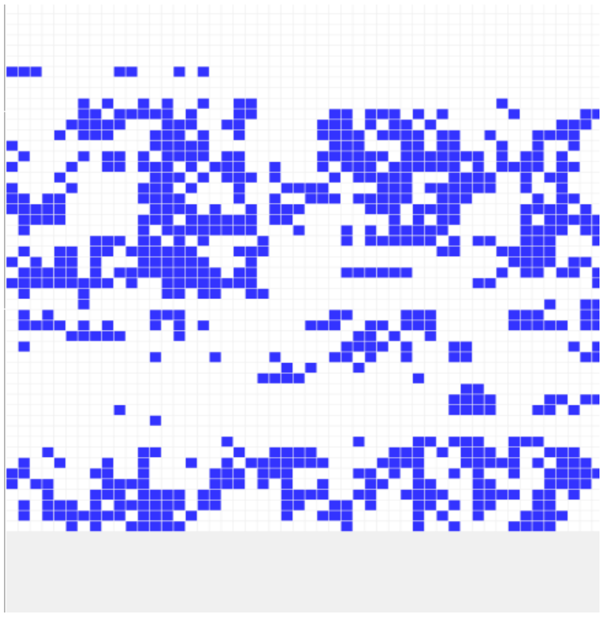


Figure 4.22 Step 2

The area of the passive cells are expanding and some of the active cells are becoming passive.

Step 3:

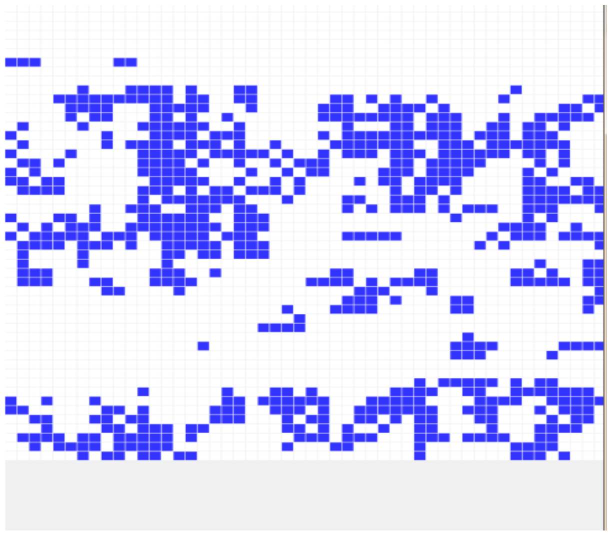


Figure 4.23 Step 3

The initial cluster formed. However, there still be some empty cells in one cluster.

Step 12:

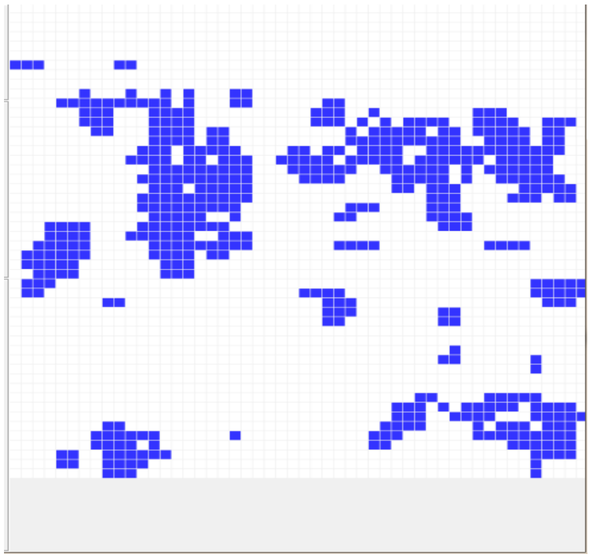


Figure 4.24 Step 12

The clear cluster is formed. We can see there are five big clusters: left, right, top, down and the center.

Step 21:

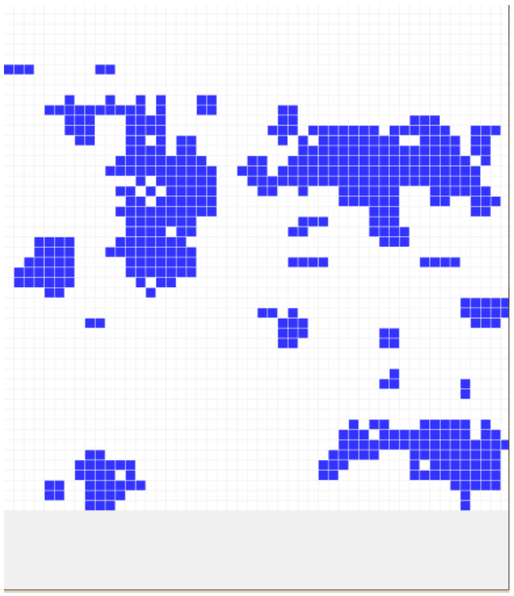


Figure 4.25 Step 21

The cluster is more clearly. We can see the prototype of the cluster. However, there are still some empty cells in the cluster.

Step 62:

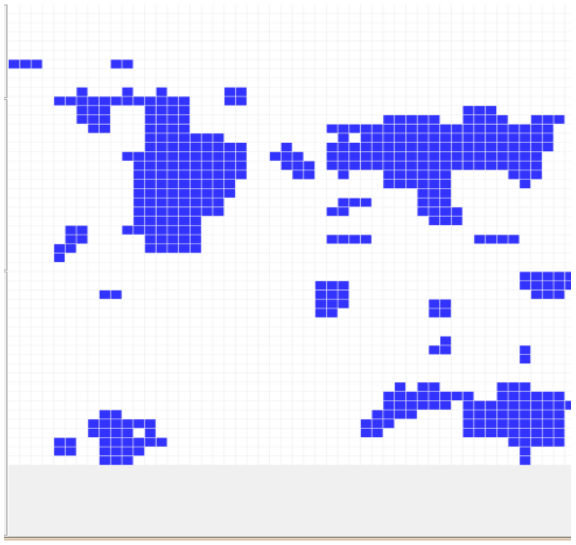


Figure 4.26 Step 62

The final cluster formed. However, there are still some empty cells. The reason for those empty cells is because every step is done with a proper probability. Therefore, there is reasonable that some empty cell in the cluster is exited.

4.4 The Detail Explain Of the Initial Cell-DEVS

The detail explanation of the advanced Cell-DEVS is as follows:

(take Pe=0.45 as an example)

Step 0:

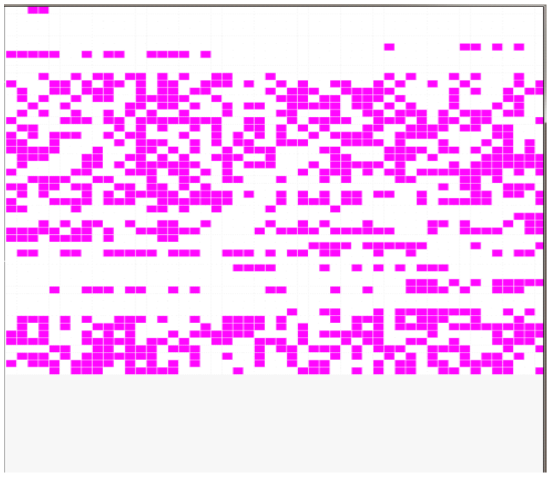


Figure 4.27 Step 0

The initial state is just the same, because the val file is not changed.

Step 1:

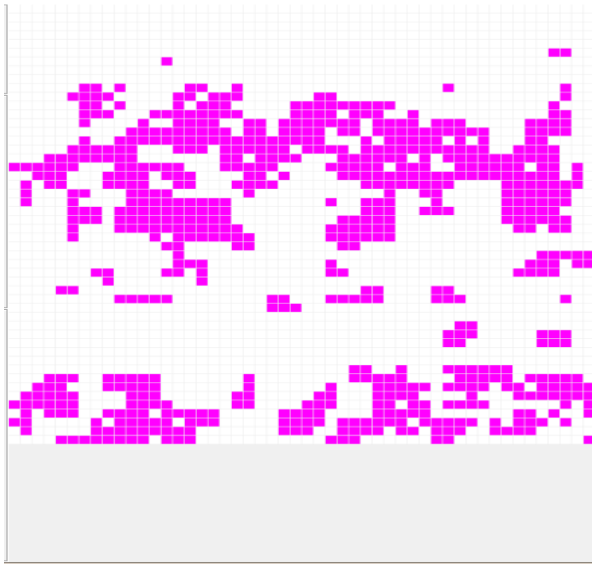


Figure 4.28 Step 1

With the new simulator , we can see that the cluster formed more quickly. Just one step can we see the obvious cluster.

Step 3:

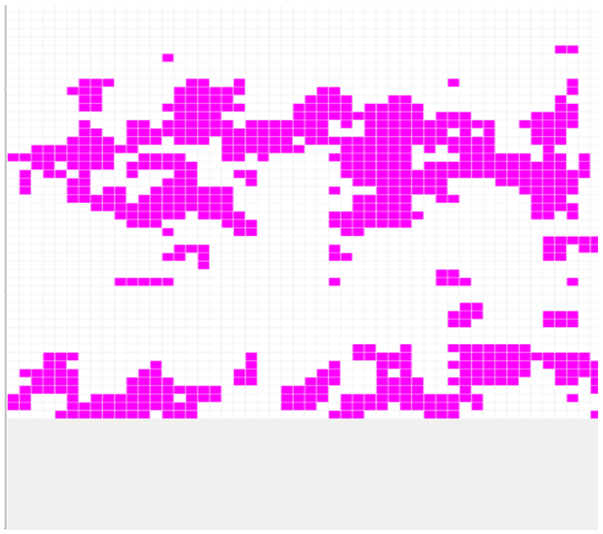


Figure 4.29 Step 3

In step 3, the initial cluster has been formed, only some small empty cells are exited.

Step 8:

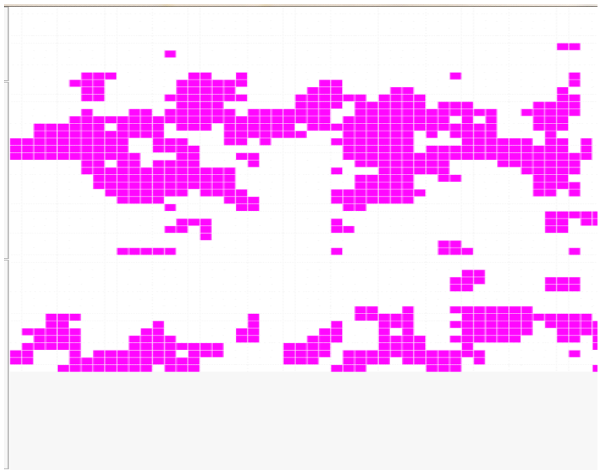


Figure 4.30 Step 8

The step 8 is the last step. From them we can see that the cluster is not as obvious as in the initial Cell-DEVS model. It is because there exits some cells that are unchangeable. Those unchangeable cells make the cluster no so obvious.

5. Performance Analysis

We use different version of CD++ to test the same model. Because the old version of CD++ do no support the state variables, we just use these to versions to implement the initial Cell-DEVS model.

The comparison is as follows:

Pe	Step	Time
0.3	13	130
0.4	32	320
0.45	62	620
0.455	93	930
0.46	59	590
0.47	35	350
0.485	65	650
0.5	32	320

Figure 5.1 the old version of CD++

The column of the step and time is as follows:

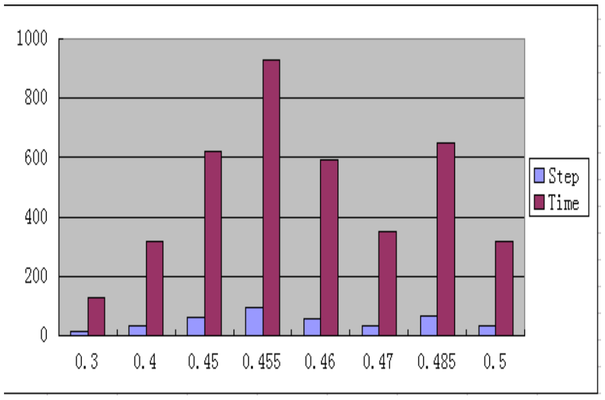


Figure 5.2 the column of new version CD++

Seeing from the column, we can find out that the time waves with the increase of Pe.When Pe equals 0.455, it takes the most of time. Also, when Pe equals 0.455, the cluster is the most obvious.

The new vision of CD++ can support state variables. However, in order to compare the performance of these two versions, we use the same rules in the two version of CD++.

The new version of CD++ is implement as follows:

Pe	Step	Time
0.3	1	10
0.4	3	30
0.45	8	80
0.455	7	70
0.46	5	50
0.47	3	30
0.485	10	100
0.5	3	30

Figure 5.3 the new version of CD++

The column of the step and time is as follows:

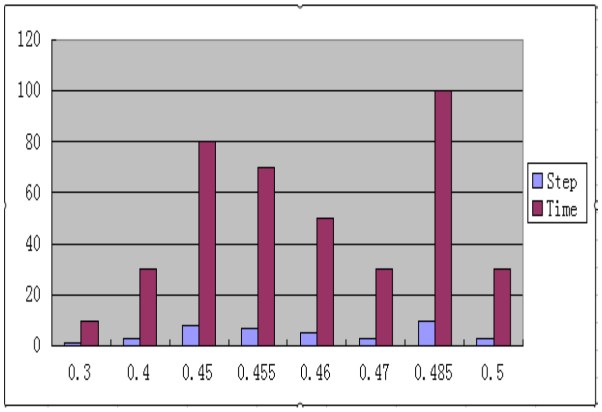


Figure 5.4 the column of new version CD++

From the column, we can clearly know that, the time still waves with the increase of Pe. However, it is when Pe=0.485 that takes the most time.

In order to have a more clear view, we make another two tables and two columns that put the results of the old version CD++ and the new version CD++ together.

The comparison for the steps of the old version CD++ and the new version CD++ is as follows:

Pe	Old Version	New Version
0.3	13	1
0.4	32	3
0.45	62	8
0.455	93	7
0.46	59	5
0.47	35	3
0.485	65	10
0.5	32	3

Figure 5.5 the table of step comparison

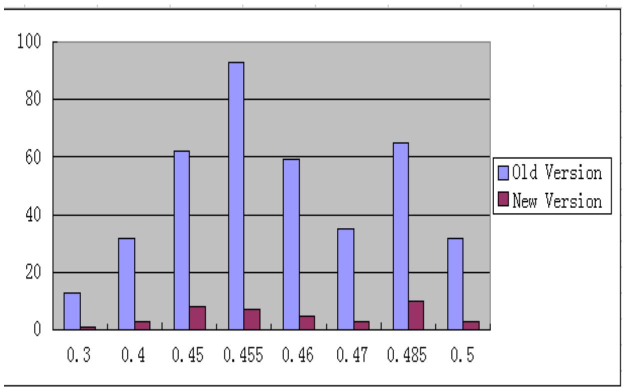


Figure 5.6 the column of step comparison

The time is:

Pe	Old Version	New Version
0.3	130	10
0.4	320	30
0.45	620	80
0.455	930	70
0.46	590	50
0.47	350	30
0.485	650	100
0.5	320	30

Figure5.7 the table of time comparison

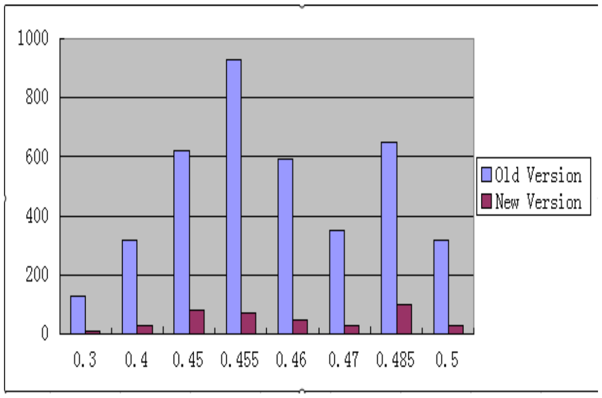


Figure 5.8 the column of time comparison

Seeing from the table and column above we can get an obvious result that the new version runs much faster than the old version(about 10 times). Also, the new version requires less steps than the old version.

Because we cannot get the simulation time in the old version, thus we cannot get a quantitative comparison.

However, as a user, we can clearly get the feeling that the new version of CD++ simulates much faster than the old version.

Therefore, supporting state variables, requiring less steps, taking less simulation time and running much faster, the new version of CD++ performs much better than the old version.

6. Conclusion

We have discussed the usage of Cell-DEVS model. And find that it is quite useful to use Cell-DEVS model to simulate different kinds of events happened in our daily life.

By using Cell-DEVS model, we can decrease the fund using in the real world; implement some difficult events which can rarely be implemented in our daily life; get some idealized results and can be used with large number of cells.

In this example presented in this paper, we actually model the following things:

1.Modeling the initial Cell-DEVS model without state variables.

This model is actually for comparing and it has been done before. By using this model, we can compare with the other model with state variables and with new version of CD++.

In the initial Cell-DEVS model, we can find two obvious phenomena. One is that with the increase of Pe(the probability that one active cell can changed its neighborhoods into active), there will be more and more active cells.

The other phenomena is that all the active cells get together forming a large cluster. This phenomena is called herding behavior. The herding behavior shows that in the real stock market, the active traders tend to be together so that they can do business better.

2.Modeling the advanced Cell-DEVS model with state variables.

The new version of CD++ allows us to use state variables for modeling. Therefore, we can make the Cell-DEVS model more close to the real world. In fact, in the real stock market, different characters of different people should be considered. Some people will be changed easily while some people cannot be changed.

By bringing in the state variables, our model is much more advanced. By comparing the advanced Cell-DEVS model with the old one, we can find some differences which can help us understand better about the dynamic of stock market.

When there is not too much cell which are unchangeable, we can see the similar phenomena as the initial Cell-DEVS model.

However, as the unchangeable cells increasing, the cluster phenomena is becoming less and less obvious.

This is easy to understand. The more unchangeable cells there is, the less cells will obey the rules. In the extreme, if all the cells are unchangeable, the rules are useless.

3. Comparing the new version CD++ with the old version CD++

The new version CD++ is not only with varies functions, but also with quite better performance.

Because the old version CD++ do not support the state variables, we just use the initial Cell-DEVS model to make the comparison.

By comparing the new version CD++ with the old one in the steps and running time, we can easily get the conclusion that the new version CD++ has a much better performance than the old version CD++.

It runs much quickly (about 10 times) than the old version CD++ and uses much less steps.

7. Reference

[1] Arthru W Brian, Steven N Durlauf, David A Lane , The Economy as an Evolving Coplex System, *Reading-Addition Wesley*, 1997

[2] wikipedia, Defination about Cell-DEVS Automata, http://en.wikipedia.org/wiki/Cellular_automaton

[3] Von Neumann, The general and logical theory of Automata, New York: Wiley, 1951

[4] Ying Shangjun, Wei Yiming, Cai Sijing, The Application of Cell-DEVS in the economy, *China Management Science*, 2000

[5] Alejandro López, Gabriel Wainer, Extending CD++ Specification Language for Cell-DEVS Model Definition , 2012

[6] Sixuan Wang, Gabriel A. Wainer, RISE User ' s Guild Manual (Integrated version) , 2012

[7] M. Bartolozzi, A. W. Thomas, Stochastic Cellular Automata Model for Stock Market Dynamics, *CSSM,* 2005
[8] B. Mandelbrot, Fractals and Scaling in Finance, Springer, New York, 1997.

[9] Wu Qingfeng，Kong Lingjiang，Influence of person's character upon the evaluation of the Cellular Automata Model for public opinion, 2005