Modeling Discrete Event System Using Cell-DEVS

(2014 Fall)

Assignment 2 – Track Combining Simulation with rule-184 in 2D Cellular Automaton Model

Student name: Zheng Xia

Student number: 7356005

University of Ottawa

Introduction

1. Problem Statement

Traffic problems have been widely concerned in recent years as the various automobiles are used in different occasions. In terms of the ways of studying the problem, the cellular automaton (also short for CA) have been implemented dramatically in simulation of traffic systems with the advantage of flexibility and simplicity. A prototype describing the traffic flow roughly in one-dimensional CA model is the rule-184 recalled in the Wolfram. In this rule, cars and roads have been regarded as different particles and discrete lattices respectively, as show in the Fig 1. All particles can go ahead at the same time in each time step as long as the next sites of theirs in the direction of advance are empty. This model, elaborately, illustrates a boost of phase transition between free movement phase and jammed phase.

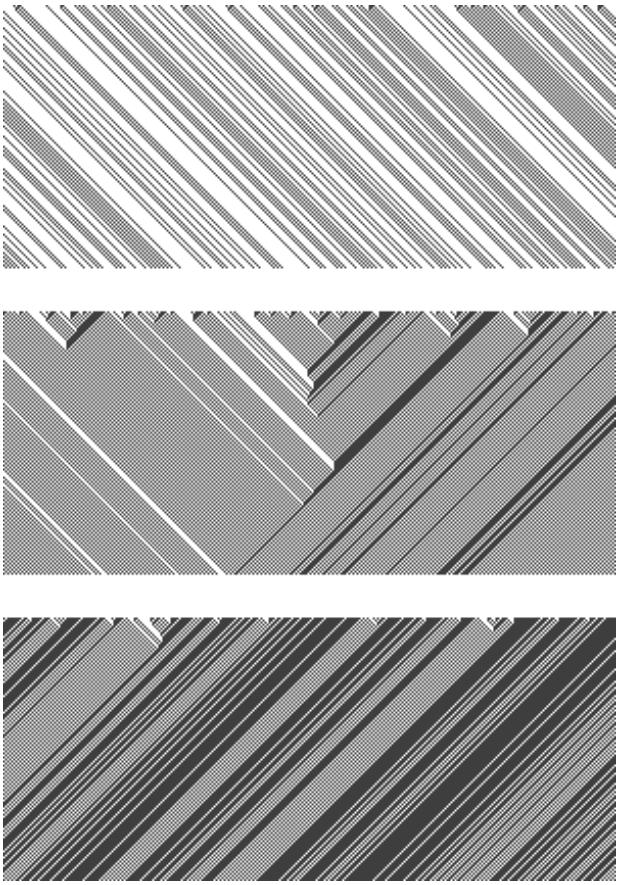

Fig 1. Rule-184 CA model simulation (after 128 steps in different conditions and density)

In this assignment, a model of track combining concept would be presented in the first place as a model of discrete event. Then, the formal model specification will be described following with rule-184 CA.

The experiment steps would be:

a) Understanding the rule-184 CA rules
b) Defined the track combining model
c) To simulate the rule-184 and the combining model
d) Test with different initial values.

Model specification

In this part, a formal specification of the traffic flow Cell-DEVS model will be presented first, followed by the rule of this model which has been illustrated in the research paper [1]. After the rules are being set, rules of track combining will also be applied in given inputs. In addition, graphical outputs would also be shown and discussed in terms of the different outcomes in initial values and the rules. Last but not least, the conclusion will be drawn as the analysis of this cellular automaton model.

Formal specification of the Rule-184 Cell-DEVS models

The more concrete Cell-DEVS formalism with port specifications is as follows:

$$CD = < X, Y, I, S, q, N, d, d_{int}, d_{ext}, t, l, D >$$

- X: Set of input External Events
- Y: Set of output External Events
- I**:** $I = < \eta, \mu, P^x, P^y >$ is the model's modular interface, with **n** neighboring cells
    - 3 total neighbors in this case for each cell
        - Neighbors: traf(0,-1), traf(0,0), traf(0,1)

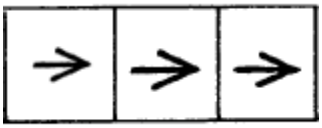

Fig 2. cell traf(0,-1)(*left*), traf(0,0)(*center*) and traf(0,1)(*right*) and the direction(arrow)

- S: possible state {0,1}
    - 0 means the cell is empty
    - 1 means the cell is occupied by a car
- The delay of the cell is 100 ms

Formal specification of the Track Combining Cell-DEVS models

By adjusting the model similarly as shown above, the neighbor for the 1st row would be:

- Neighbors: traf(0,-1), traf(0,0), traf(0,1), traf(1,0)

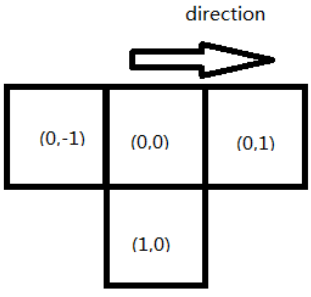

Fig 3. cell traf(0,-1)(*left*), traf(0,0)(*center*), traf(0,1)(*right*), traf(1,0)(*bottom*) and the direction(arrow) on 1st row

Which represent the front, back cell and the cell below itself. The 2nd row has five neighbors representing the front, back cell, the cell above itself and the cell right above would be:

- Neighbors: traf(0,-1), traf(0,0), traf(0,1), traf(-1,0), traf(-1,1)

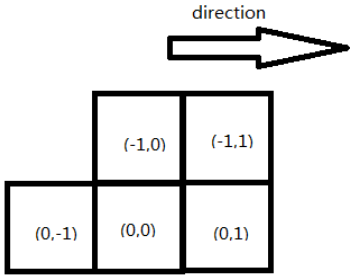

Fig 3. cell traf(0,-1)(*left*), traf(0,0)(*center*), traf(0,1)(*right*), traf(-1,0)(*above*), traf(-1,1) and the direction(arrow) on 2nd row

Others are the same as that defined in Rule-184 CA model

Rules of reaction in Cell-DEVS model

Rules in the Cellular Automaton model define the general behavior of the cell through the time sequence. That is, it describes how the occupation situation of the neighbor of the cell affect the cell value after the delay time delay. In the tracking combining process, the rules will show how the car from 1st row move down to the 2nd drive lane.

- Rule-184

This rule has been widely used in the model of traffic flow simulation, especially the single lane of a highway. By applying this rule, vehicles (representing as particles) move in a single direction and the moving procedure is controlled by the car in front of them. The number of car in the simulation can remain unchanged [2]. Noticed that the rules have the same coding methodology as the Wolfram Cellular Automaton model. The rule describes that the model is deterministic. That is, there is no acceleration in the car moving process (which means the all the cars have only one speed, i.e. $V_{max} = 1$)

rule : 1 100 { (0,-1) = 1 and (0,0) = 1 and (0,1) = 1}
rule : 0 100 { (0,-1) = 1 and (0,0) = 1 and (0,1) = 0}
rule : 1 100 { (0,-1) = 1 and (0,0) = 0 and (0,1) = 1}
rule : 1 100 { (0,-1) = 1 and (0,0) = 0 and (0,1) = 0}
rule : 1 100 { (0,-1) = 0 and (0,0) = 1 and (0,1) = 1}
rule : 0 100 { (0,-1) = 0 and (0,0) = 1 and (0,1) = 0}
rule : 0 100 { (0,-1) = 0 and (0,0) = 0 and (0,1) = 1}
rule : 0 100 { (0,-1) = 0 and (0,0) = 0 and (0,1) = 0}

- Rules in track combination

By combining the rule-184 methods and the car moving procedure, the new rules would be shown as follows:

%1--first row
rule : 0 100 { (1,0)=0 and (0,0)=1 and cellpos(0)=0 } %move down

rule : 0 100 { (1,0)=1 and (0,-1)=0 and (0,0)=0 and (0,1)=0 and cellpos(0)=0 } %no car

rule : 0 100 { (1,0)=1 and (0,-1)=0 and (0,0)=0 and (0,1)=1 and cellpos(0)=0 } %car ahead
rule : 0 100 { (1,0)=1 and (0,-1)=0 and (0,0)=1 and (0,1)=0 and cellpos(0)=0 } %move ahead
rule : 1 100 { (1,0)=1 and (0,-1)=0 and (0,0)=1 and (0,1)=1 and cellpos(0)=0 } %stuck

rule : 1 100 { (1,0)=1 and (0,-1)=1 and (1,-1)=1 and (0,0)=0 and cellpos(0)=0 } %car behind
rule : 0 100 { (1,0)=1 and (0,-1)=1 and (1,-1)=1 and (0,0)=1 and (0,1)=0 and cellpos(0)=0 } %car behind
rule : 1 100 { (1,0)=1 and (0,-1)=1 and (1,-1)=1 and (0,0)=1 and (0,1)=1 and cellpos(0)=0 } %car behind

%2--second row
%car coming from above
rule : 1 100 { (-1,0)=1 and (0,0)=0 and cellpos(0)=1 } %car insert

%if the cell is occupied
rule : 1 100 { (-1,0)=1 and (0,0)=1 and (-1,1)=1 and cellpos(0)=1 } %car stay because there is a car on right above
rule : 0 100 { (-1,0)=1 and (0,0)=1 and (-1,1)=0 and (0,-1)=0 and (0,1)=0 and cellpos(0)=1 } %move ahead
rule : 1 100 { (-1,0)=1 and (0,0)=1 and (-1,1)=0 and (0,-1)=0 and (0,1)=1 and cellpos(0)=1 } %car stay
rule : 0 100 { (-1,0)=1 and (0,0)=1 and (-1,1)=0 and (0,-1)=1 and (0,1)=0 and cellpos(0)=1 } %move ahead
rule : 1 100 { (-1,0)=1 and (0,0)=1 and (-1,1)=0 and (0,-1)=1 and (0,1)=1 and cellpos(0)=1 } %move ahead

%if there is no car above but there is car on the above right
rule : 0 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=0 and (0,0)=0 and (0,1)=0 and cellpos(0)=1 } %no car
rule : 0 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=0 and (0,0)=0 and (0,1)=1 and cellpos(0)=1 } %no car
rule : 1 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=0 and (0,0)=1 and (0,1)=0 and cellpos(0)=1 } %car stay
rule : 1 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=0 and (0,0)=1 and (0,1)=1 and cellpos(0)=1 } %car stay
rule : 1 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=1 and (0,0)=0 and (0,1)=0 and cellpos(0)=1 } %car behind
rule : 1 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=1 and (0,0)=0 and (0,1)=1 and cellpos(0)=1 } %car behind
rule : 1 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=1 and (0,0)=1 and (0,1)=0 and cellpos(0)=1 } %move ahead
rule : 1 100 { (-1,0)=0 and (-1,1)=1 and (0,-1)=1 and (0,0)=1 and (0,1)=1 and cellpos(0)=1 } %car stay

```
%if there is no car above and above right
rule : 0 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=0 and (0,0)=0 and (0,1)=0 and
cellpos(0)=1 } %no car
rule : 0 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=0 and (0,0)=0 and (0,1)=1 and
cellpos(0)=1 } %no car
rule : 0 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=0 and (0,0)=1 and (0,1)=0 and
cellpos(0)=1 } %move ahead
rule : 1 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=0 and (0,0)=1 and (0,1)=1 and
cellpos(0)=1 } %stay
rule : 1 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=1 and (0,0)=0 and (0,1)=0 and
cellpos(0)=1 } %car behind
rule : 1 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=1 and (0,0)=0 and (0,1)=1 and
cellpos(0)=1 } %car behind
rule : 0 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=1 and (0,0)=1 and (0,1)=0 and
cellpos(0)=1 } %move ahead
rule : 1 100 { (-1,0)=0 and (-1,1)=0 and (0,-1)=1 and (0,0)=1 and (0,1)=1 and
cellpos(0)=1 } %car stay

rule : 0 100 { t }
```

Through the changing, a prospective view of the moving procedure can be described. That is, after the delay, the high priority of the car from 1st row can move down to the 2nd row as soon as there exist any space in the 2nd row which will mean the car in the 1st row would move out as quick as they can, otherwise it would go ahead until they meet the boarder. In turns, if the car in the 2nd row detects a car in the front above, the car would not move and will stay in its position instead.

The rest of rules describe the inner rule of each row, which is the implementation of the rule-184. Specifically, the neighbor behind and it finds the room in front is empty, both its neighbor and itself can move ahead. That is, in the case of 110 in traf(0,-1), traf(0,0), traf(0,1), the current cell with would still be occupied because of the car sequence in the next time step. In this way the original car will move out and the new car will come after the delay, which also could represent a moving of a set of cars from a typical traffic model in the real world.

Graphical Output Result

In this session, graphical result will be shown with different rules implementation. In addition, different test cases will be used to compare in the next step.

- Rule-184
    - Initial value of row: 11111010110001101010

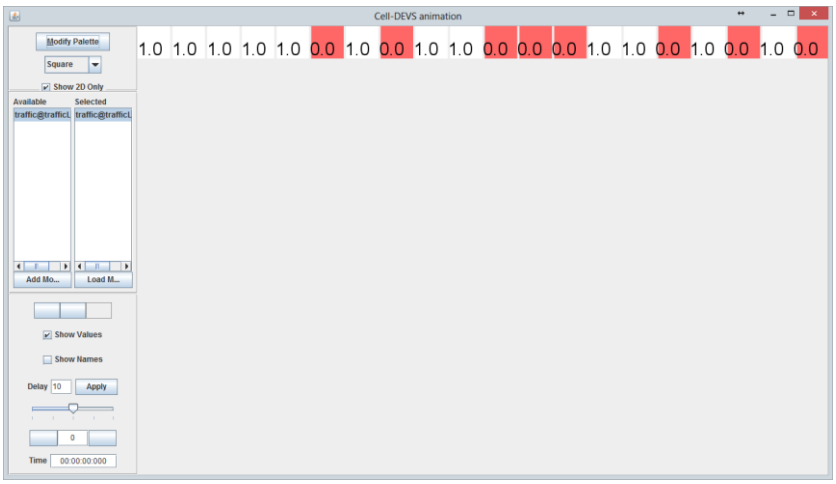

Fig 4. Initial state (time = 0ms)

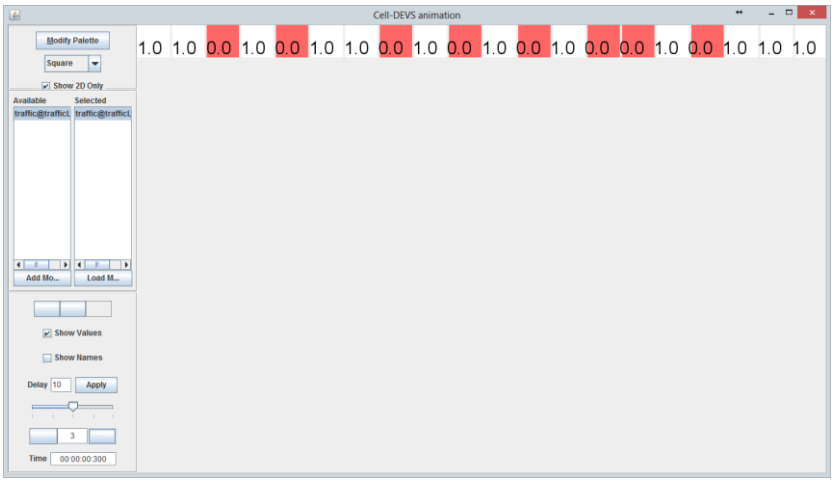

Fig 5. Three-frames state (time = 300ms)

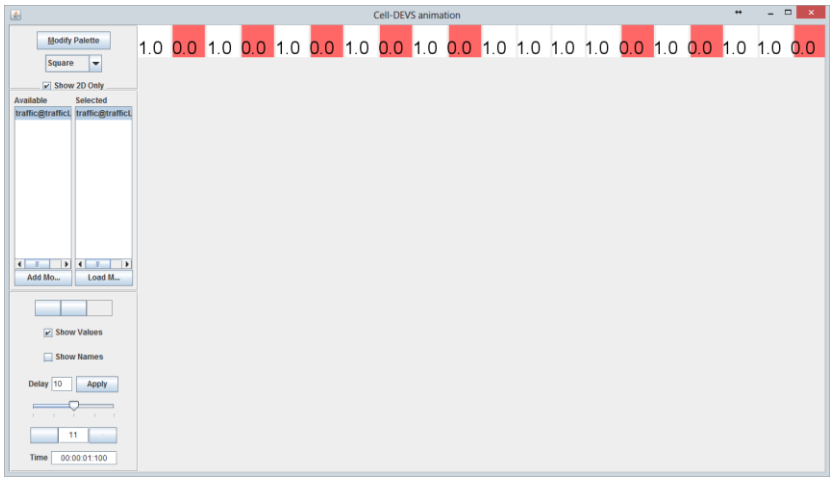

Fig 6. Eleven-frames state (time = 1100ms)

Fig 7. Thirteen-frames state (time = 1300ms)

- Track combining rules

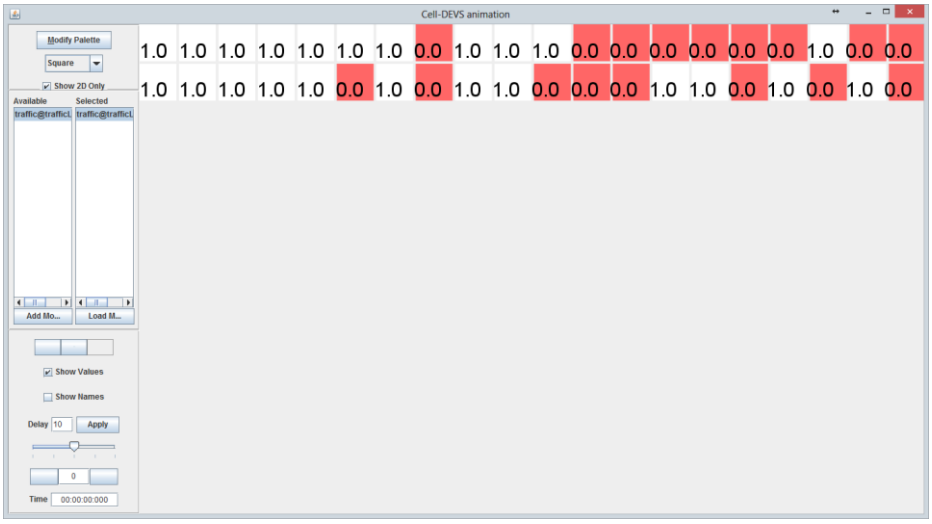

Fig 8. An example of track combing rules (time = 0ms)

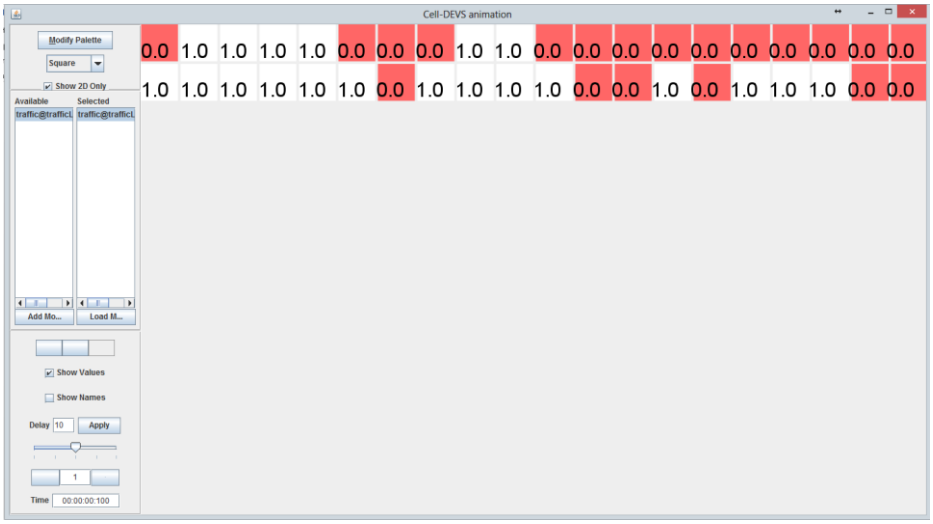

Fig 9. An example of track combing rules (time = 100ms)

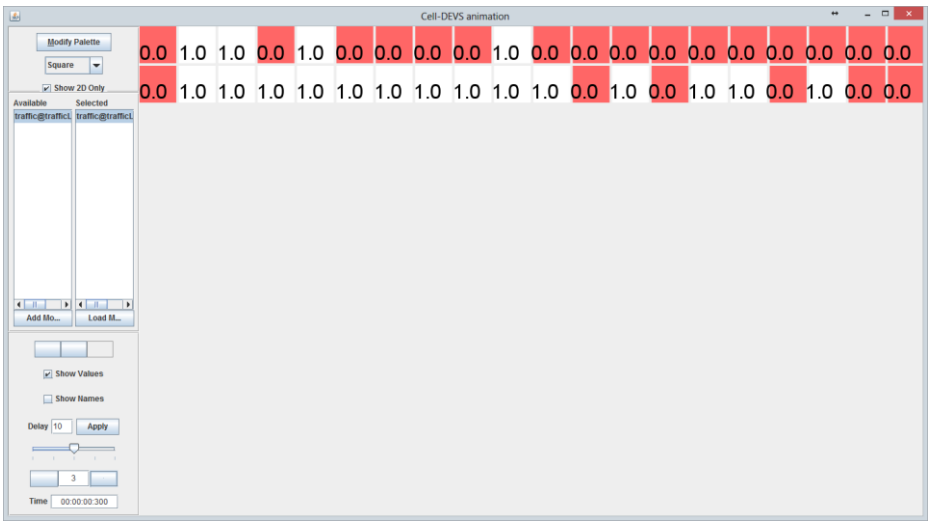

Fig 9. An example of track combing rules (time = 300ms)

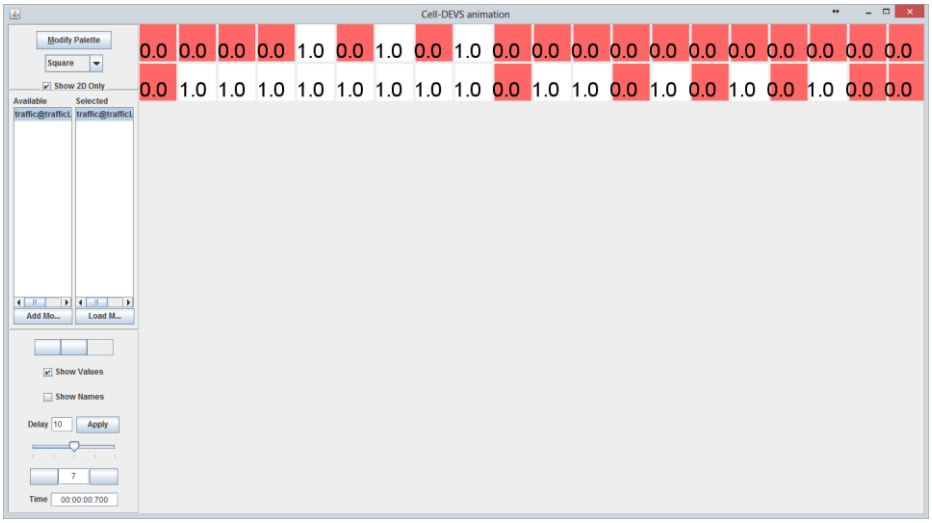

Fig 9. An example of track combing rules (time = 700ms)

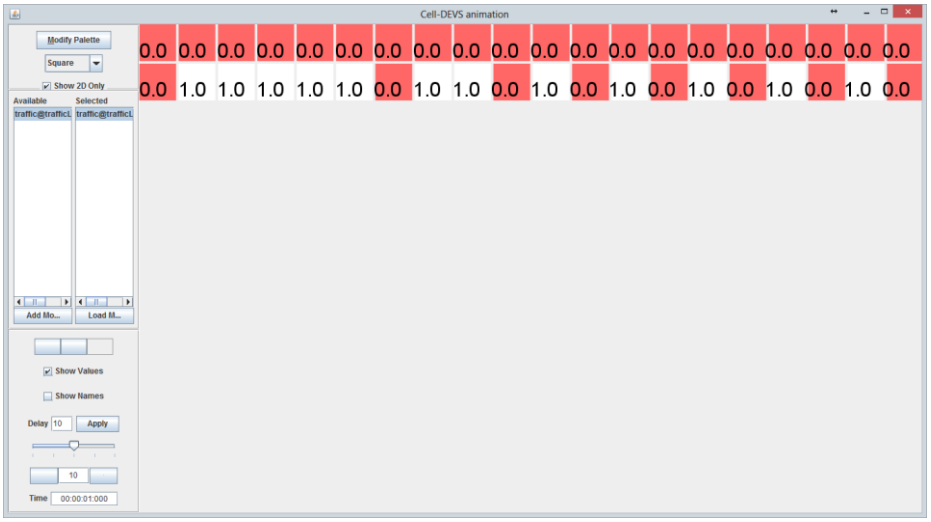

Fig 10. An example of track combing rules (time = 1000ms)

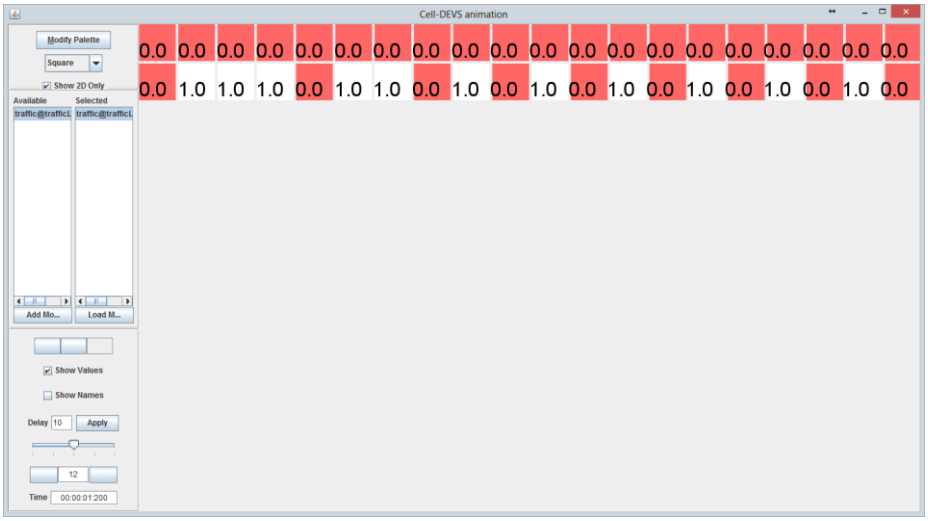

Fig 11. An example of track combing rules (time = 1200ms)

Conclusion

As we can see from the graphical result, the track combining model has been successfully implemented via this Cell-DEVS model using rule. This result indicates the car leaving 1st for 2nd can be described precisely as the rules of priority have been illustrated in the traffic-rule in the lane moving process, yet there are some tidy problems about the cars' defining when the cars come to the boarder, which might be the undefined in CD++.

Reference

[1]. Fukui, Minoru; Ishibashi, Yoshihiro.Traffic Flow in 1D Cellular Automaton Model Including Cars Moving with High Speed[J]. Journal of the Physical Society of Japan, 1996, 65(6):1868

[2]. http://en.wikipedia.org/wiki/Rule_184