**SYSC 5104 METHODOLOGIES FOR DISCRETE EVENT MODELLING AND SIMULATION**

**Assignment 2 – Data mining with Cellular Automata**

Yasser Jafer

Ottawa University – Student Number: 2554284

[yjafe089@uottawa.ca](mailto:yjafe089@uottawa.ca)

**Part I**

**Conceptual Model:**

Data mining in general term is the process of extracting patterns from data and one main step in data mining is referred to as data mining classification. There are several techniques used for data mining classification, including nearest neighbour, decision tree learning and support vector machines.

The cells in a CA are individually very simple but together they can represent complex behaviours and they are capable of self-organization. The goal of this assignment is to explore the use of cellular automata for data mining and more specifically classification. The referenced paper is (T.Fawcett, “Data mining with cellular automata”, SIGDKK Explorations, Vol10, Issue 1, P32-39.).

The intention is to show 2 dimensional classification of similar classes. A simple voting rule was used for this purpose. Based on the paper a CA update rule called n4\_V1 was used and this rule examines each cell’s four neighbours and sets its class to the majority class. This way each cell in the grid becomes assigned with the class of it’s nearest neighbour.

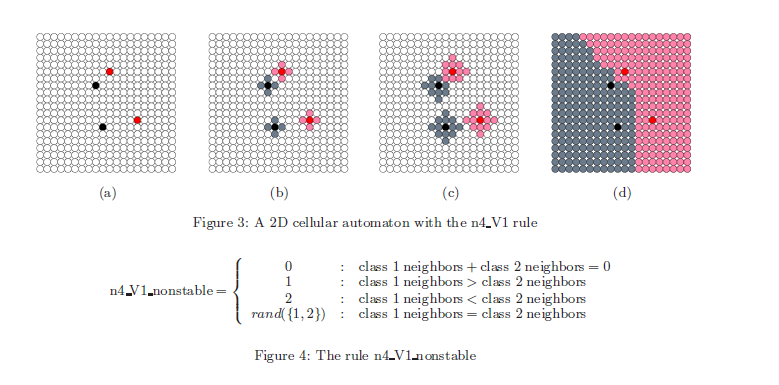
1. To show how cellular automata rules can represent data mining classification using Neumann neighbourhood having two classes 1, 2.



1. We want to see how adding extra class i.e. class 3 will affect the classification results and to proof the Neumann neighbourhood is more scalable compared with Moore neighbourhood rules as the paper suggests.

**Part A: 2 Class datamining – model described in the paper:**

The paper sets forward the following rules called n4\_V1\_nonstable and the rule is described below:



(Fawcett, 2008)

Formal Cell-Dev Specifications:

The following is the formal specification for the Cell-DEVS 2class datamining model

CD = < X, Y, I, S, θ, N, d, δint, δext, τ, λ, D >

X = Ø

Y = Ø

S = {0, 1, 2}

N = neighbourhood = Von Neumann

{(-1,0), (0,-1), (0,0), (0,1), (1,0)}

d = 100 ms

τ: N🡪S:

S: //rules as shown in datamining\_2class.ma.

rule : 1 100 {statecount(1)>statecount(2) }

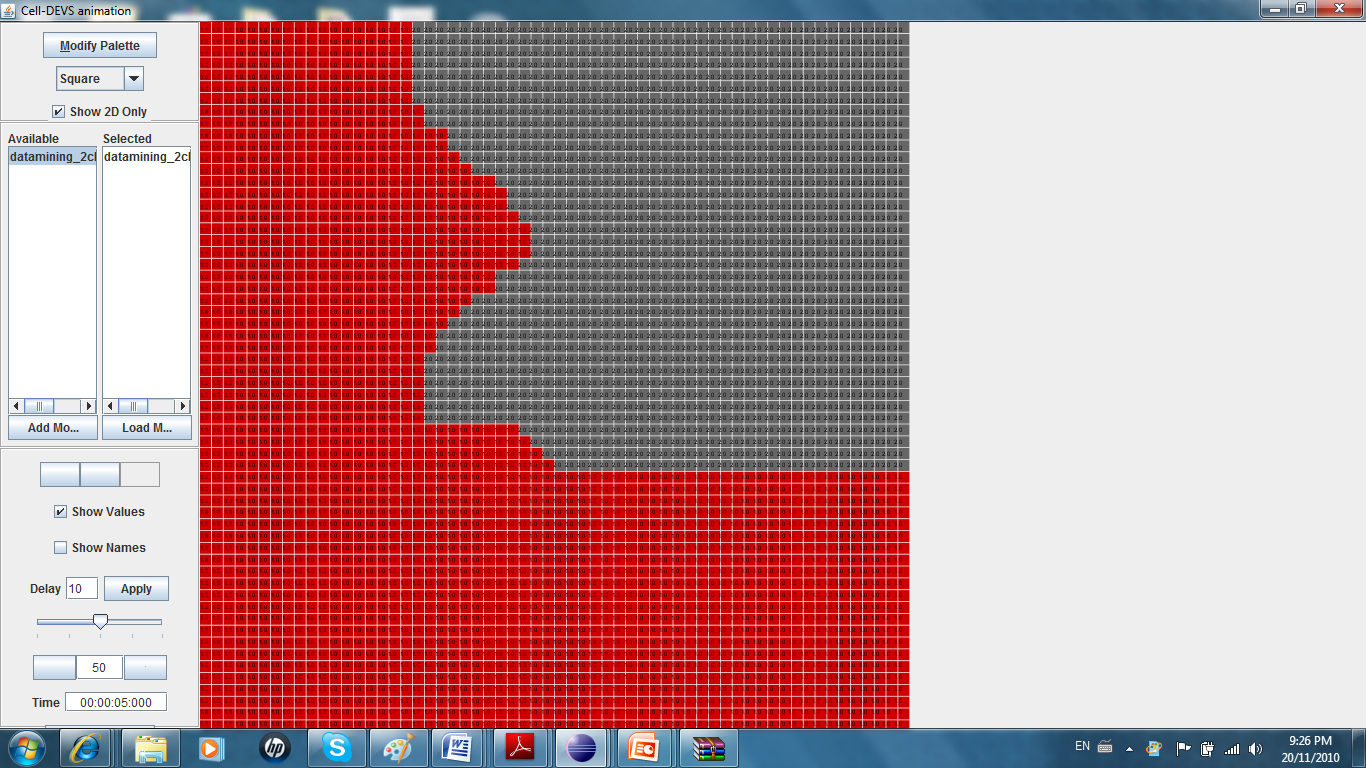
rule : 2 100 {statecount(2)>statecount(1) }

rule : 0 100 {statecount(1)=0 and statecount(2)=0 }

rule : {if(uniform(0,1)>0.5,1,2)} 100 { statecount(1)=statecount(2) }

rule : {(0,0)} 100 { t }

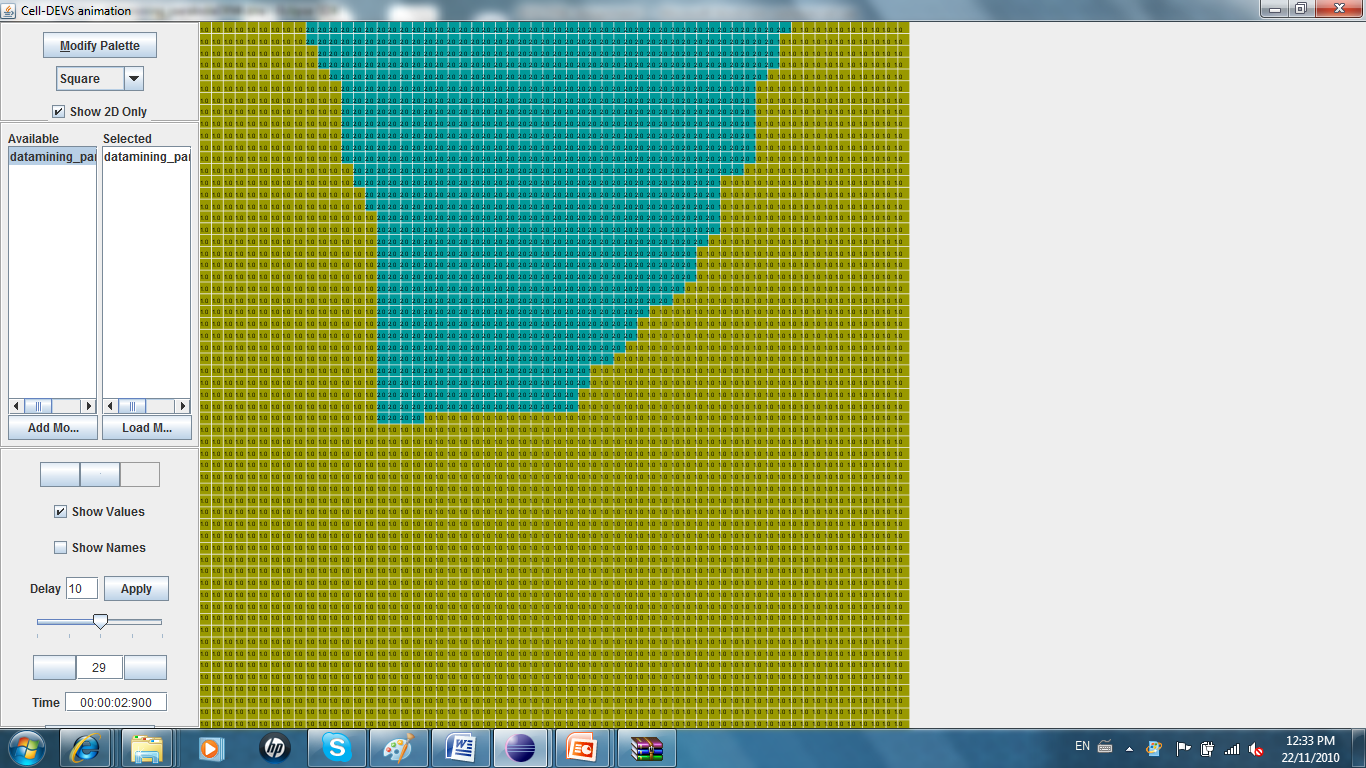
We have selected 60x60 with two classes 1 and 2 as shown in figure (a) below:

****

**Figure a) Graphical output of 60x60 dimensio**

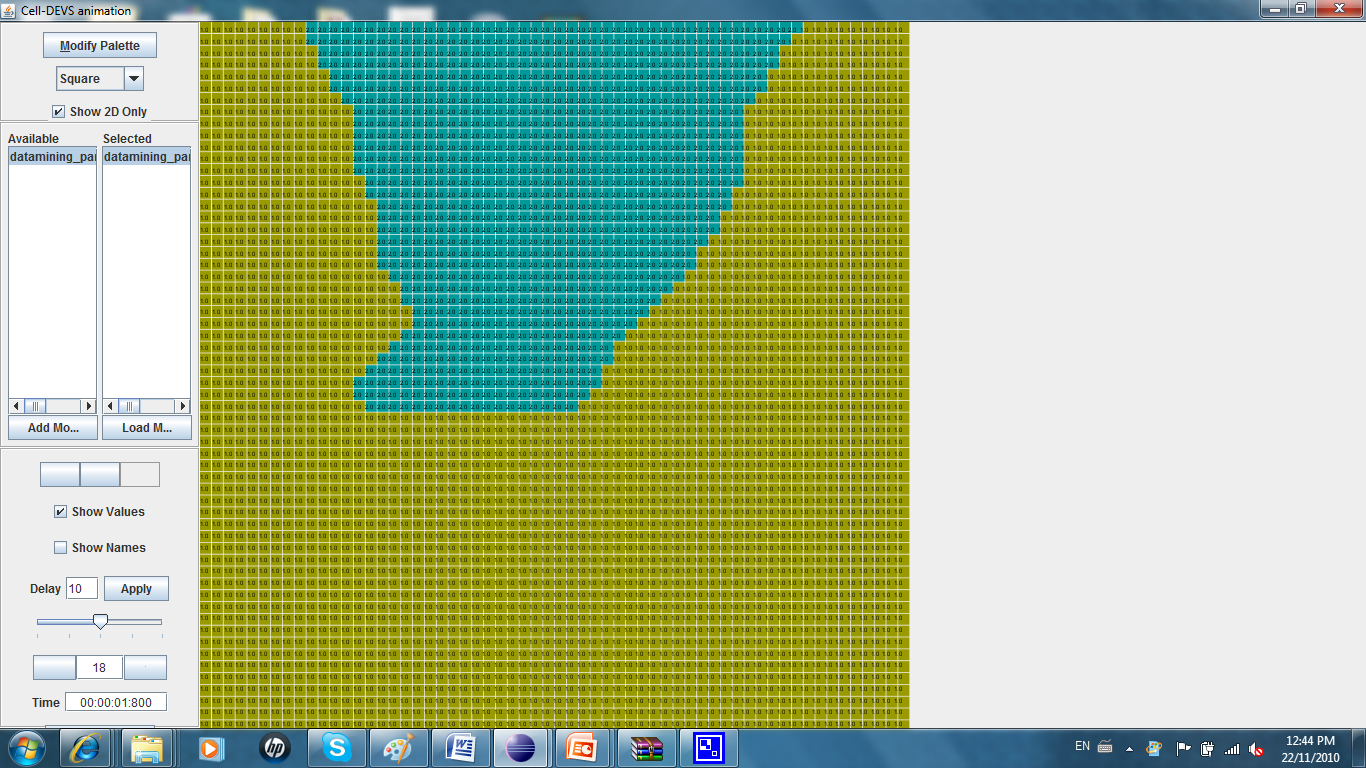
We’ve tested two parabola model, one using Neumann and another using Moore neighbourhood, and since we know the actual shape of parabola, it seems that for the same distribution of original cells of class 1 and class 2 using Neumann neighbourhood is giving better results (more realistic results) compared with Moore neighbourhood . The results were illustrated as follows:

**Part A) b) Parabola: Von Neumann neighbourhood**

****

**Figure (b) Graphical output – von Neumann neighbourhood**

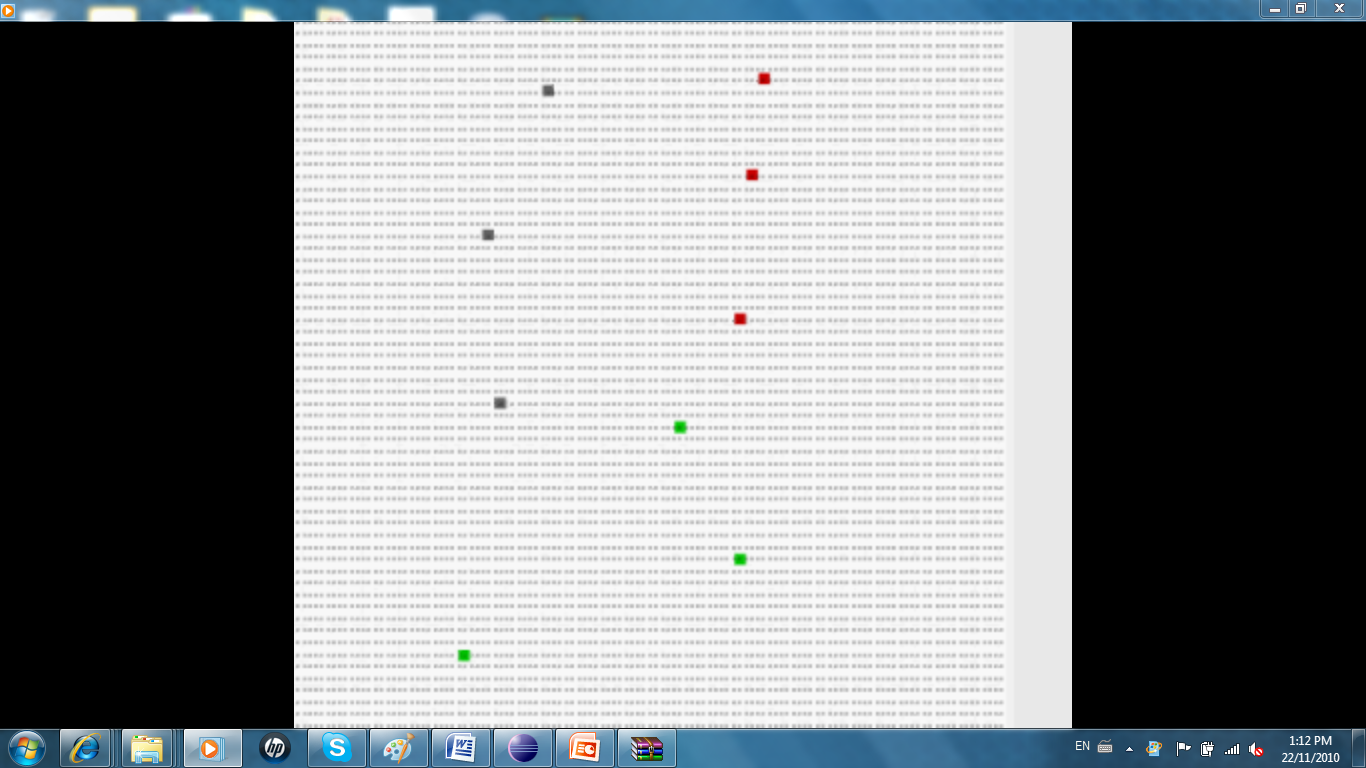
**Part A)b) Parablola: Moore neighbourhood**

****

**Figure (c) Graphical output – Moore neighbourhood**

**Part B)A) 3 Class Example (Improvement of paper example)**

The following is the formal specification for the Cell-DEVS 3class datamining model. For each class 3 (each class is represented in a color) original cells were selected positioned as shown in the following graphical output:



Formal Cell-Dev Specifications:

CD = < X, Y, I, S, θ, N, d, δint, δext, τ, λ, D >

X = Ø

Y = Ø

S = {0, 1, 2, 3}

N = neighbourhood = Von Neumann

{(-1,0), (0,-1), (0,0), (0,1), (1,0)}

d = 100 ms

τ: N🡪S:

S: //rules as shown in datamining\_3class.ma. The rules

rule : 1 100 {statecount(1)> statecount(2) and statecount(1)> statecount(3)}

rule : 2 100 {statecount(2)> statecount(1) and statecount(2)> statecount(3)}

rule : 3 100 {statecount(3)> statecount(1) and statecount(3)> statecount(2)}

rule : 1 100 {statecount(2)> statecount(3) and statecount(2)< statecount(1)}

rule : 1 100 {statecount(3)> statecount(2) and statecount(3)< statecount(1)}

rule : 2 100 {statecount(1)> statecount(3) and statecount(1)< statecount(2)}

rule : 2 100 {statecount(3)> statecount(1) and statecount(3)< statecount(2)}

rule : 3 100 {statecount(1)> statecount(2) and statecount(1)< statecount(3)}

rule : 3 100 {statecount(2)> statecount(1) and statecount(2)< statecount(3)}

rule : 0 100 {statecount(1)=0 and statecount(2)=0 and statecount(3)=0}

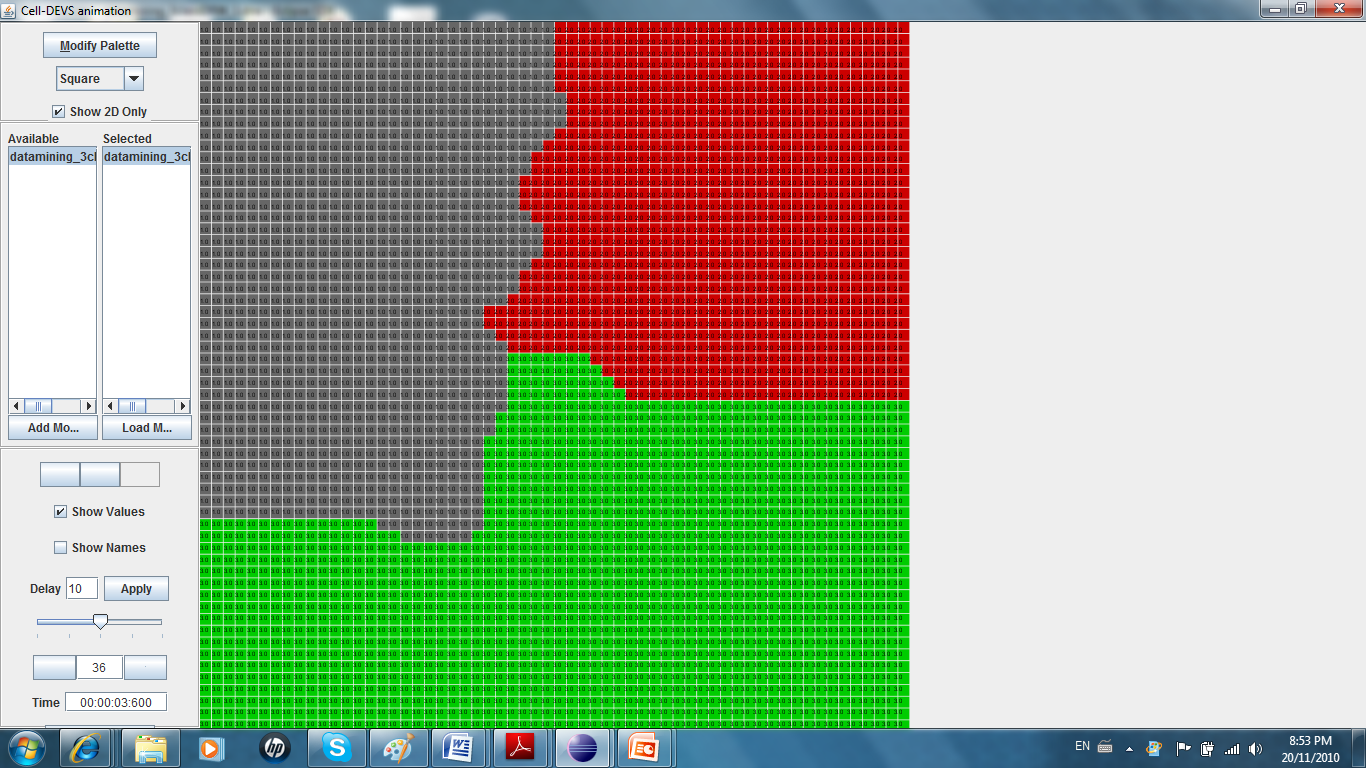
rule : {if(uniform(0,1)>0.66,1,if(uniform(0,1)>0.5,2,3))} 100 { statecount(1)=statecount(2) and statecount(2)=statecount(3) }

rule : {if(uniform(0,1)>0.5,1,2)} 100 {statecount(1)=statecount(2)}

rule : {if(uniform(0,1)>0.5,2,3)} 100 {statecount(2)=statecount(3)}

rule : {if(uniform(0,1)>0.5,1,3)} 100 {statecount(1)=statecount(3)}

rule : {(0,0)} 100 { t }

****

**Figure (b) Graphical output of 3 class datamining (60x60) dimension**

**Part B)B) Class Example – Moore neighbourhood and comparing the results**

Formal Cell-Dev Specifications:

The following is the formal specification for the Cell-DEVS 3class datamining model

CD = < X, Y, I, S, θ, N, d, δint, δext, τ, λ, D >

X = Ø

Y = Ø

S = {0, 1, 2, 3}

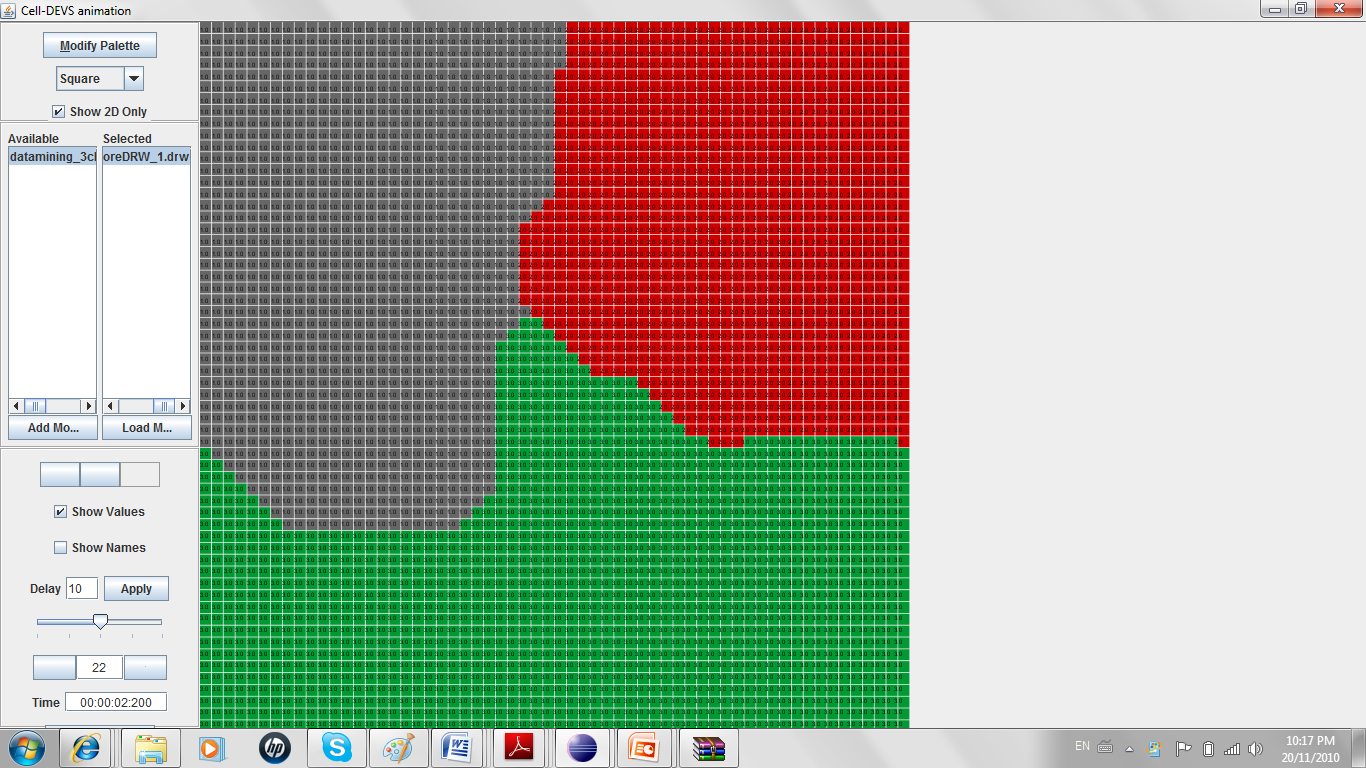
N = neighbourhood = Moore

{(-1,0), (0,-1), (0,0), (0,1), (1,0), (-1,-1), (1,1), (1,-1), (-1,1)}

d = 100 ms

τ: N🡪S:

S: //rules as shown in datamining\_3class\_moore.ma.

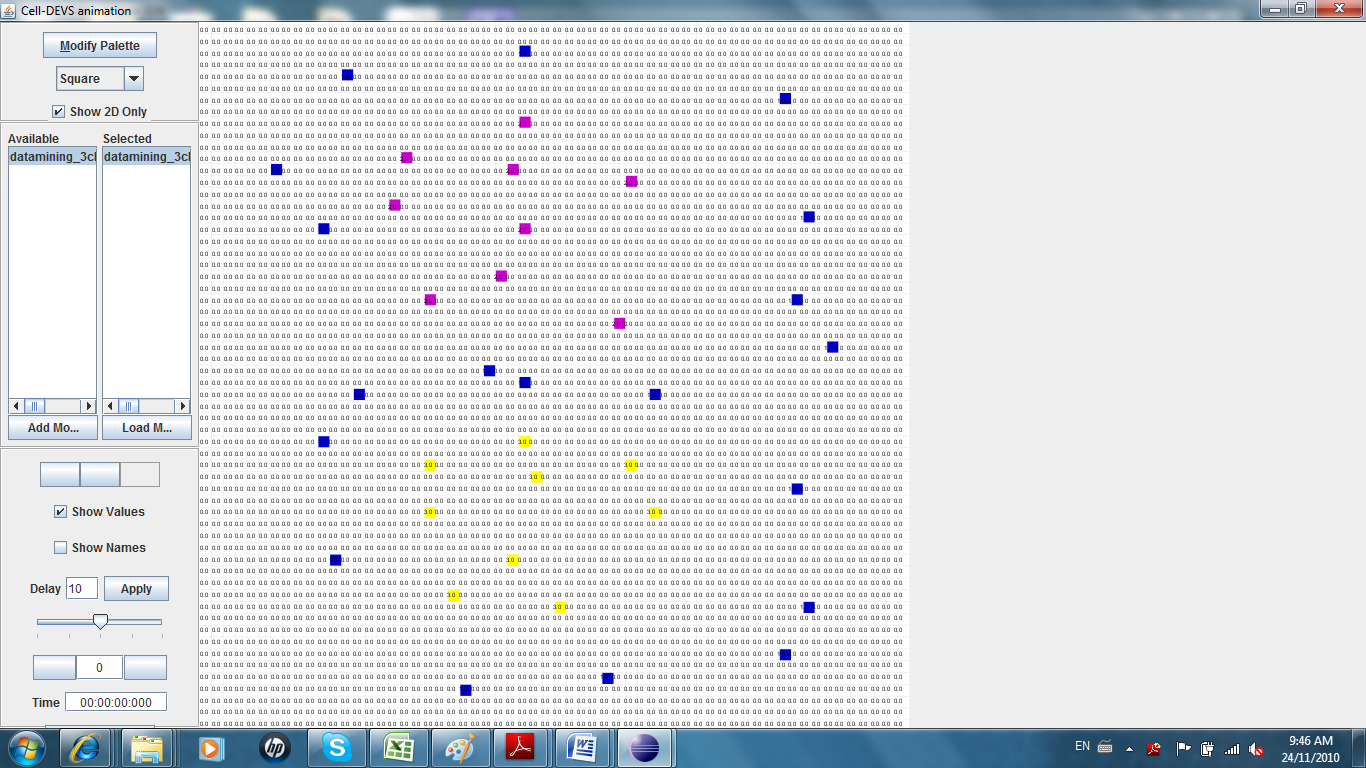


**Figure (c) Graphical output of 3 class datamining using Moore neighbourhood (60x60 dimension)**

Paper states that it is using von Neumann neighbourhood because it is linear in the number of dimensions of the instance space, so it scales well. It was shown in the parabola case that choosing von Neumann neighbourhood tends to have more similarity to the real parabola. In case of 3 classes it is difficult to make such a comparison.

**Part C) A) 3 Class example – ambient (von Neumann Neighbourhood)**

In here we will test the ambient class distribution, where 2 classes are circulated by another class:



and the results are shown as follows:

Formal Cell-Dev Specifications:

CD = < X, Y, I, S, θ, N, d, δint, δext, τ, λ, D >

X = Ø

Y = Ø

S = {0, 1, 2, 3}

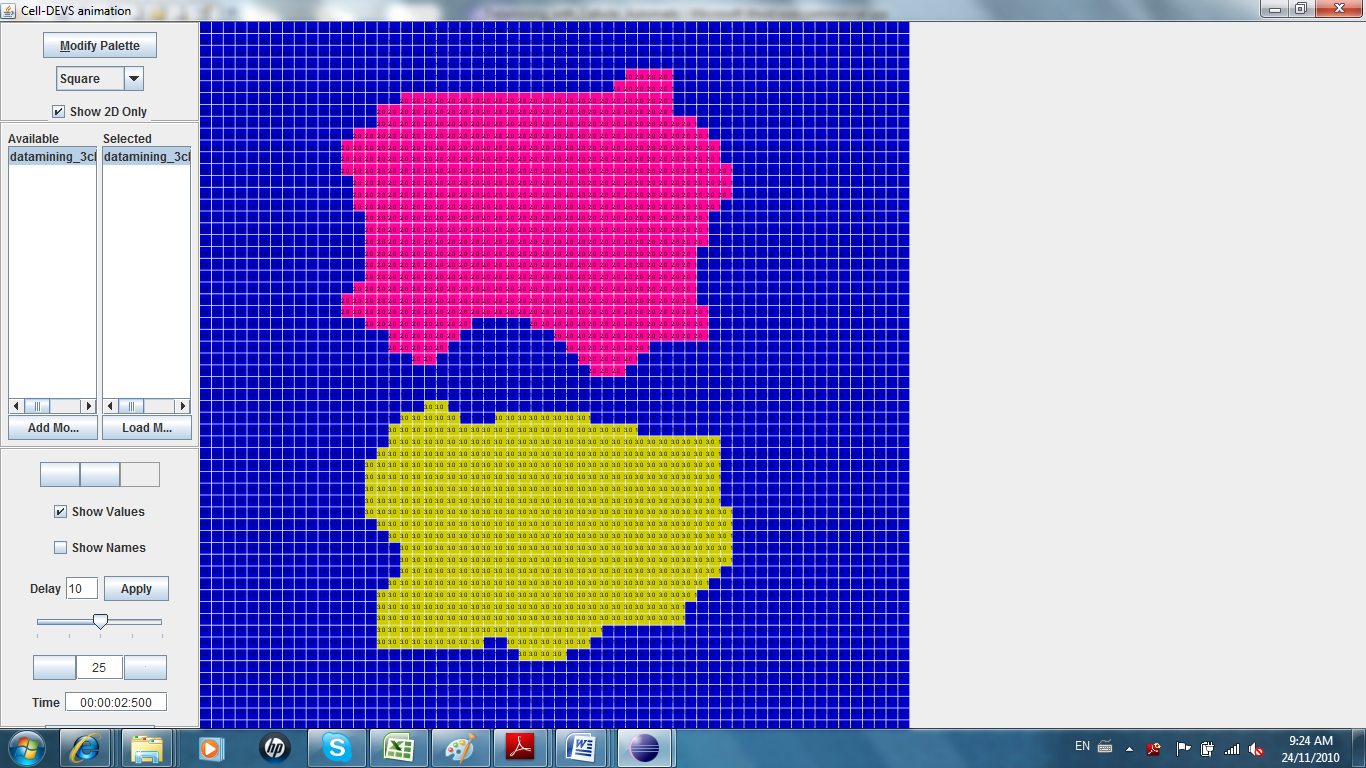
N = neighbourhood = Von Neumann

{(-1,0), (0,-1), (0,0), (0,1), (1,0)}

d = 100 ms

τ: N🡪S:

S//As before



**Figure (d) Graphical output of 3 class ambient datamining (60x60) dimension**

**Part C) B) 3Class example – ambient (Moore Neighbourhood)**

Formal Specification:

CD = < X, Y, I, S, θ, N, d, δint, δext, τ, λ, D >

X = Ø

Y = Ø

S = {0, 1, 2, 3}

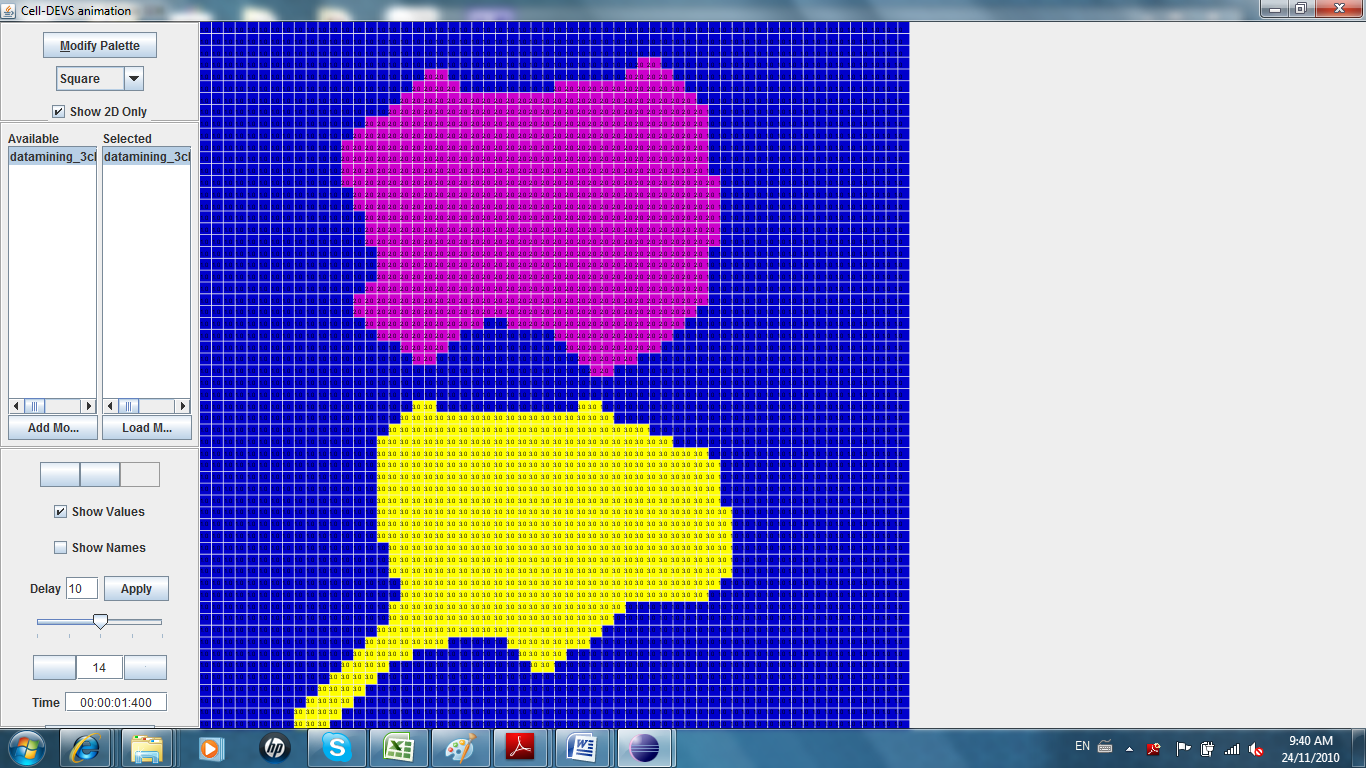
N = neighbourhood = Moore

{(-1,0), (0,-1), (0,0), (0,1), (1,0), (-1,-1), (1,1), (1,-1), (-1,1)}

d = 100 ms

τ: N🡪S:

S: //As before

****

**Figure (d) Graphical output of 3 class ambient datamining (60x60) dimension – Moore neighbourhood**

For the same set of initial cells we get Figure (d), when Moore neighbourhood, this example and the parabola example both are in line with what the paper was suggesting that von Nuemann neighbourhood is more scalable for data mining classification.