Two Cellular Automaton Models for Reaction-Diffusion Systems: Theory and Simulation

by

Lingyu Yi

B.Sc., Inner Mongolia University, 2002

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN THE DEPARTMENT
OF
MATHEMATICS

SIMON FRASER UNIVERSITY
Fall 2006

APPROVAL

Name: Lingyu Yi

Degree: Master of Science

Title of thesis: Two Cellular Automaton Models for Reaction-Diffusion Systems: Theory and Simulation

Examining Committee: Dr. Ralf Wittenberg
Chair

Dr. Robert Russell
Senior Supervisor

Dr. Steve Ruuth
Supervisor

Dr. Adam Oberman
Supervisor

Dr. JF Williams
External Examiner

Date Approved: July 28, 2006

Abstract

A cellular automaton (CA) is a discrete microscopic dynamical system widely used to investigate and understand the mechanisms of complex systems such as reaction-diffusion systems based on cell-cell interactions. We introduce two CA models for Turing-type pattern formation. These are a moving average CA and lattice-gas CA. For a moving average CA, the construction of the local CA rules from the reaction-diffusion partial differential equations relies on a moving-average procedure to implement the diffusive step and a probabilistic table lookup for the reactive step. We apply this method to the 2D Brusselator system. The corresponding 11-state CA model is able to capture the Hopf and Turing birfucation. For a lattice-gas CA, we introduce a modified reaction rule for an activator-inhibitor system and combine it with the propagation rule and shuffling rule. A variety of dynamics arise in this LGCA model. Numerical simulations of both CA models are presented and analyzed.

Acknowledgments

First I would like to express my deepest gratitude and sincere appreciation to my thesis advisor and mentor Dr. Robert Russell for his invaluable guidance, patience and encouragement. Absolutely, I would not have been able to accomplish this work without him. All of his critics, suggestions, comments and corrections, which have finely shaped my character, gave me the motivation to excel in what I am doing despite the inevitable difficulties. His broad knowledge, research enthusiasm and deep insights have been a very important resource and inspiration to my accomplishment. I am very proud of being his student and I hope to continue learning from his spirit on both the personal and scientific levels throughout my life.

I would like to thank my supervisory committee members: Dr. Steve Ruuth and Dr. Adam Oberman for their feedbacks and helpful comments. Also acknowledge helps and encouragements provided by Chair Dr. Ralf Wittenberg.

Special thanks to Dr. JF Williams for his guidance, encouragement and helpful comments.

Special thanks to my generous and warm-hearted friends Colin and Ruonan. They shared their precious experience with me and taught me a lot during my study and work.

I would also like to thank everyone at SFU who helped me in the past three years, specially the Department of Mathematics.

Dedication

To my parents and lovely sister.

Contents

List of Figures

Chapter 1

Introduction

A cellular automaton (CA) is a discrete dynamical system which consists of a collection of cells on a lattice of specified shape that evolves through a number of discrete time steps according to a set of rules based on the states of neighboring cells. First introduced by von Neumann in the early 1950s to develop an abstract model of self-reproduction in biology, CA have been widely developed and used to model different complex systems in physics, computer science, biology, etc.. Comprehensive studies of cellular automata have been performed by Wolfram starting in the 1980s, and his fundamental research in the field can be found in [2]. One of the most interesting features of CA modeling is that the complexity and the global behavior of a system emerge just from local interactions of cells following simple local rules.

The CA is also used to simulate and investigate reaction-diffusion systems ([1][7][8]), which provides a way to investigate and analyze the spatio-temporal dynamics, especially the Turing pattern formation, at a microscopic level. In 1952, Turing pointed out that diffusion plays a very important role in the loss of stability of a spatially homogeneous stable steady state of a reactive system. This type of instability is called *Turing instability.* A spatially heterogeneous pattern may arise in this situation, which is usually referred to as Turing pattern formation. Turing-type pattern (Diffusion-driven pattern) formation can be well captured and interpreted by macroscopic continuous models such as reaction-diffusion partial differential equations (PDEs) ([4]). Different from PDEs, a CA is a discrete model in space, time and state. It is characterized by

the lattice $\mathcal{L}$, the state set $\mathcal{E}$, the interaction neighborhood $\mathcal{N}$ and the evolution rule $\mathcal{C}$. In the CA modeling, the reaction and diffusion process are simulated by introducing different evolution rules with the help of $\mathcal{L}$, $\mathcal{E}$, and $\mathcal{N}$. In the framework of CA, the patterns can be seen as the emergent behaviors of the system only due to local interactions between cells following local rules. The CA provides an alternative way instead of a replacement of PDEs approach to investigate and analyze Turing pattern formation problems at a microscopic level.

In this thesis, we firstly apply a so-called Moving average CA (MACA) method to simulate a two dimensional activator-substrate system, the Brusselator. To mimic the diffusion process, we average the local sum in a Moore neighborhood or an extended-Moore interaction neighborhood using a very efficient algorithm called the moving average method ([8]). A reaction rule is obtained from the nonlinear functions in the PDEs. A probabilistic truncation rule is also introduced to make the states of each cell represented by integers. A two-dimensional two-component CA model for the Brusselator is then constructed based on these rules. This CA model can be seen as a special finite-difference discretization to the PDEs which only involves integer operations. Numerical simulations show that this CA model is able to capture the Hopf instability and Turing instability and is qualitatively correct when compared with the results obtained by an IMEX finite-difference scheme, involving second order backward differentiation formula (2-SBDF). We secondly investigate the Turing pattern formation in reaction-diffusion systems using another type of CA, the lattice gas CA (LGCA). With the help of *channels* introduced for each cell, the diffusion process can be realized by combining the Propagation rule $\mathcal{P}$ and Shuffling rule $\mathcal{M}$ ([1] [7] [10]). We then design a Reaction rule $\mathcal{R}$ which is based on the rule for an activator-inhibitor system described in [1]. Compared with the original rule, this modified rule $\mathcal{R}$ has more control of the LGCA system and is able to generate more complex and interesting behaviors and patterns. We focus on how a LGCA model for an activator-inhibitor model based on rule $\mathcal{R}$ is characterized by the lattice $\mathcal{L}$, state $\mathcal{E}$ and interaction neighborhood $\mathcal{N}$. For $\mathcal{L}$, we start with a one-dimensional lattice and then extend to a two-dimensional lattice. For $\mathcal{E}$, we investigate the difference

between a LGCA model with a rest channel and a LGCA model without a rest channel, i.e., the role of a rest channel in the LGCA model for a Turing pattern formation problem. Different $\mathcal{N}$ are introduced for the same model in one- and two-dimensions and the corresponding dynamics are simulated and analyzed. This LGCA model also shows qualitatively similar results to those observed in a reaction-diffusion system modeled by PDEs. We compare this LGCA model with the Gray-Scott model in one-dimension. Similar structures such as self-reproduction, a traveling pulse, and a standing pulse arise in the LGCA model. The commutativity of the three rules $\mathcal{P}$, $\mathcal{M}$ and $\mathcal{R}$ is also investigated through numerical simulations.

The layout of this thesis is as follows. In Chapter 2, we briefly review the basic ideas and some classical examples of CA and Turing pattern formation. In Chapter 3, we investigate a two-dimensional activator-substrate model, the Brusselator, using the MACA method and the 2-SBDF method. In Chapter 4, a LGCA model for an activator-inhibitor system is constructed based on a modified reaction rule. Different dynamics are simulated and discussed. Comparison with the one-dimension Gray-Scott model is given and analyzed. The commutativity of the evolution operators $\mathcal{P}$, $\mathcal{M}$ and $\mathcal{R}$ is also studied. Conclusions are made in Chapter 5.

Chapter 2

Cellular Automata and Pattern Formation

In this chapter, we briefly review the main ideas and some classical examples of cellular automata and Turing pattern formation. Most of this material is taken from the standard references [1] [2] [3] [4].

2.1 Cellular Automata

In general, a cellular automaton (CA) is specified by the following definition (see for instance [1]),

- $\mathcal{L}$: a regular discrete lattice of cells and boundary conditions,
- $\mathcal{E}$: a finite set of states that characterize the cells,
- $\mathcal{N}$: a finite set of cells that defines the interaction neighborhood of each cell, and
- $\mathcal{C}$: a rule that determines the dynamics of the states of the cells.

From the definition we can see that cellular automata can be seen as a class of discrete models. The meaning of discrete is:

Space:	They consist of a discrete one-, two- or three-dimensional spatial lattice of cells.
Time:	They evolve in discrete time steps following simple local interaction rules.
State:	Each cell only has a finite discrete set of possible values.

The CA considered in this thesis has three important features. One is the homogeneity of CA, which means that all cells are identical and equivalent. Another is the parallelism of CA evolution, i.e., each cell follows the same simple local interaction rule simultaneously at each discrete time step. The last is the locality of CA, which means that the state of each cell at time $t+1$ only depends on the states of itself and its nearby neighbors at time t.

2.1.1 Classical Cellular Automata

The simplest nontrivial CA [2] is the one-dimensional CA which consists of a line of cells, and each cell only has two states denoted by 0 or 1. At every time step there is then a definite rule that determines the state of a given cell from the states of itself and its two immediate neighbors, the left one and the right one. Mathematically, we have a one-dimensional lattice $\mathcal{L}$ which consists of L cells, $L \in \mathbb{N}$. Each cell is labeled by its position $r \in \mathcal{L}$. The state of cell r at time t is denoted by $x(r,t)$, where $x(r,t) \in \mathcal{E} = \{0,1\}$, $t \in \mathbb{N}$. $X(r,t) = (x(r-1,t), x(r,t), x(r+1,t)) \in \{0,1\}^3$, a Boolean vector, denote the state of $\mathcal{N}(r)$, where $\mathcal{N}(r) = \{r-1, r, r+1\}$ is the interaction neighborhood. There are $2^3 = 8$ possible states for a $\mathcal{N}(r)$, i.e.,

$$X(r,t) \in \{(1,1,1),(1,1,0),(1,0,1),(1,0,0),(0,1,1),(0,1,0),(0,0,1),(0,0,0)\},$$

so there are $2^8 = 256$ possible rules for determining $x(r,t+1)$, the state of cell r at time $t+1$. For example, the Rule 1 says if $X(r,t) = (0,0,0)$, then $x(r,t+1) = 1$, otherwise $x(r,t+1) = 0$. The Rule 4 says if $X(r,t) = (0,1,0)$, then $x(r,t+1) = 1$, otherwise $x(r,t+1) = 0$. Figure 2.1 shows a graphical representation of Rule 1 and

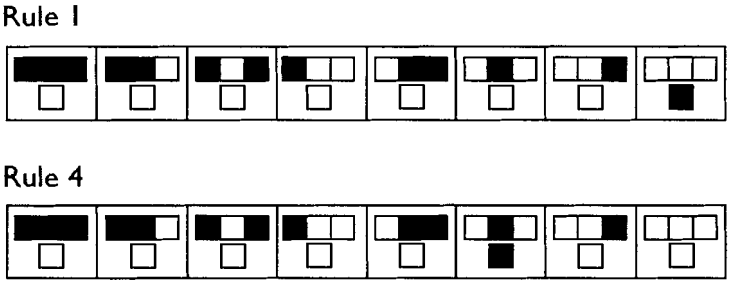

Figure 2.1: An illustration of Rule 1 and Rule 4

Rule 4, where we use black to denote 1 and white to denote 0. The numbering of the Rules is determined by the possible states of $x(r, t+1)$; for example, Rule 250 and Rule 90 are named so because in binary, 250 and 90 are written as 11111010 and 01011010, respectively, as shown in Figure 2.2. From the illustration of Rule 250 and Rule 90, the difference between them is very small, but the long time behaviors are quite different. Figure 2.3 shows that Rule 90 generates a fractal structure pattern while Figure 2.4 shows that Rule 250 generates a checkerboard structure pattern. The initial condition is a single cell started in state 1 (black), the others in 0 (white). The CA generated by Rule 1 is usually called Rule 1 CA, and so on. Another interesting pattern is generated by Rule 30 CA which shows similar pattern to some seashells, like the one in Conus and Cymbiola Genus as shown in Figure 2.5.

Despite the simple construction of CA, they are capable of generating very complex dynamics or patterns, and such dynamics only arise from the local interactions. In [2], Wolfram *qualitatively* classifies the behaviors of CA into four basic classes based on his systematical investigations of the simplest one-dimensional deterministic CA:

Class 1:	limit point
Class 2:	limit cycle
Class 3:	chaotic - "strange" attractor
Class 4:	more complex behaviour

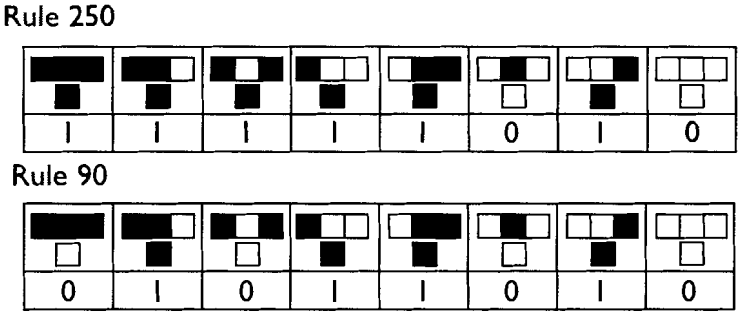

Figure 2.2: An illustration of Rule 250 and Rule 90

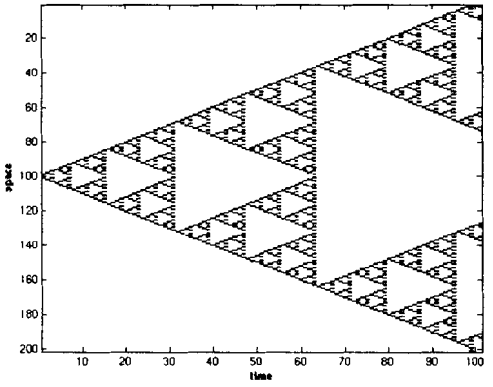

Figure 2.3: Fractal structure in CA system generated by Rule 90 after 100 time steps.

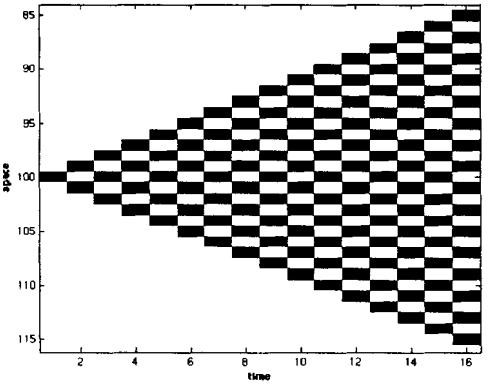

Figure 2.4: Checkerboard structure in CA system generated by Rule 250 after 16 time steps.

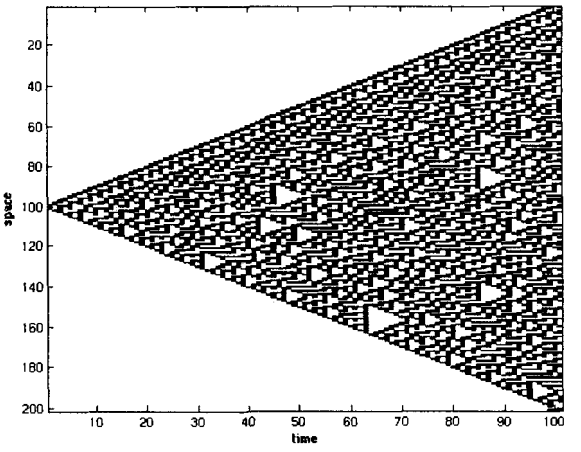

Figure 2.5: Patterns generated by Rule 30 after 100 time steps.

Class 1 cellular automata

After a finite number of time-steps, evolution leads to a homogeneous state from (nearly) all possible initial configurations.

Class 2 cellular automata

Evolution leads to either simple stable states or periodic and separated structures.

Class 3 cellular automata

From nearly all initial configurations, evolution leads to chaotic patterns which are a kind of self-similar fractal curves.

Class 4 cellular automata

After finite steps of time, evolution leads to complex, localized propagating structures.

Another example is called Conway's Game of Life which is one of the first applications showing that CA are capable of producing dynamic patterns and structures. It is a two-dimensional CA with binary cell states. The rules introduced by John Horton Conway are described by the following:

1. A cell that is dead (0) at time step t, becomes alive (1) at time $t + 1$ if exactly three of the eight neighboring cells at time t are alive.

2. A cell that is alive at time t dies at time $t + 1$ if at time t fewer than two or more

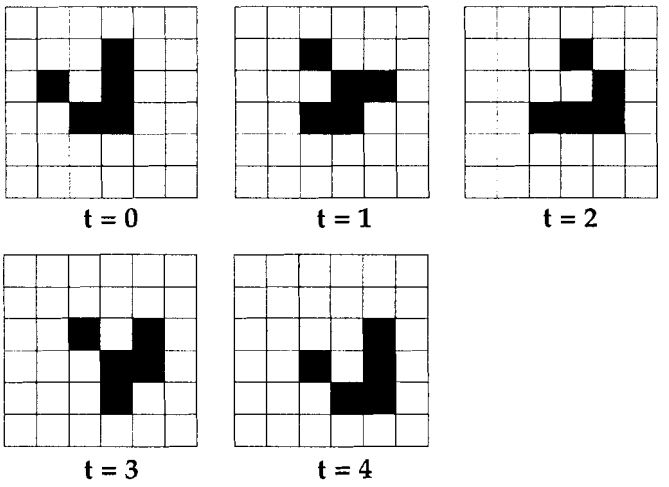

Figure 2.6: Conway's Game of Life: Glider. It consists of 5 live cells and reproduces itself in a diagonally displaced position every four time steps.

than three neighboring cells are alive.

One of the most intriguing patterns in Conway's Game of Life is a so-called glider, which shows an oscillatory propagating pattern, as seen in Figure 2.6.

2.1.2 Lattice Gas Cellular Automata

In this section, a special type of Cellular Automaton, a lattice-gas Cellular Automaton (LGCA), is introduced and discussed. The key differences between the classical CA and LGCA are the introduction of *channels* at each cell on the lattice in the LGCA system and the *propagation* of particles in the channels. In the LGCA system, each cell can have several channels and particles can only reside in channels. There are two types of channels: velocity channesl and rest (zero velocity) channels. Particle residing on a velocity channel will jump to another velocity channel according to the speed and the direction of the channel. Particle residing on rest channel will not move. The presence of a rest channel is not necessary in some LGCA models, but the rest channel plays a very important role in the pattern formation problems in the LGCA system. Without it the LGCA system for pattern formation may exhibit checkerboard structure, which is artificial and doesn't have any interpretable meaning in the real system.

In general, a LGCA is specified by the following definition (see for instance [1]):

- $\mathcal{L}$: a regular discrete lattice of cells and boundary conditions. Velocity and rest channels are associated with each cell,
- $\mathcal{E}$: a finite set of states that characterize the cells,
- $\mathcal{N}$: a finite set of cells that defines the interaction neighborhood of each cell, and
- $\mathcal{C}$: a rule that determines the dynamics of the states of the cells.

A lattice $\mathcal{L} \subset \mathbb{R}^d$ consists of a set of cells, where the cell is labeled by its position $r \in \mathcal{L}$. Each cell possesses s channels which are labeled by $(r, c_i), i = 1, ..., s$. An exclusion principle is imposed which requires that no more than one particle can reside on the same channel (r, c_i) for all $r \in \mathcal{L}, i = 1, ..., s$. Thus, each cell can possess at most s particles. The state of channel (r, c_i) at time t is denoted by $\eta_i(r, t)$ with

$$\eta_i(r,t) = \begin{cases} 1 & \text{presence of a particle in channel } (r, c_i) \text{ at time } t \\ 0 & \text{absence of a particle in channel } (r, c_i) \text{ at time } t. \end{cases}$$

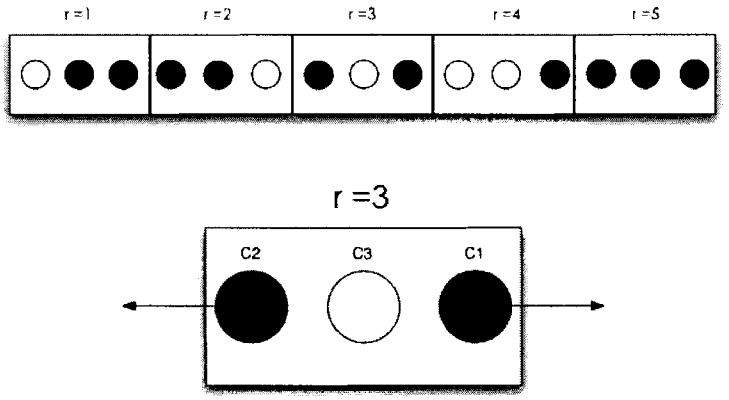

Figure 2.7: Example of a one-dimensional LGCA with five cells and two velocity channels and one rest channel on each cell. Filled dot represents the presence of a particle in the channel. Arrows represent the moving directions of a particle in that channel.

The state of a cell r at time t is then given by a Boolean vector $\eta(r,t)$ with

$$\eta(r,t) = (\eta_1(r,t), ..., \eta_s(r,t))).$$

Thus, for a LGCA the set of elementary states $\mathcal{E}$ of a single cell r is given by

$$\mathcal{E} = \{0,1\}^s \tag{2.1}$$

The total number of particles (concentration) of cell r at time t is given by

$$n(r,t) := \sum_{i=1}^{s} \eta_i(r,t).$$

Figure 2.7 shows an example of a one-dimensional LGCA which consists of five cells and three channels (two velocity channels and one rest channel) on each cell. Figures 2.8 and 2.9 give an example of a cell in a two-dimensional LGCA with five channels (four velocity channels and one rest channel) and 4 channels (4 velocity channels and no rest channel), respectively.

The time evolution of a LGCA system occurs at discrete time steps and follows from the iterated application of an evolution operator $\mathcal{C}$,

$$[state]^{t+1} = \mathcal{C}[state]^t \tag{2.2}$$

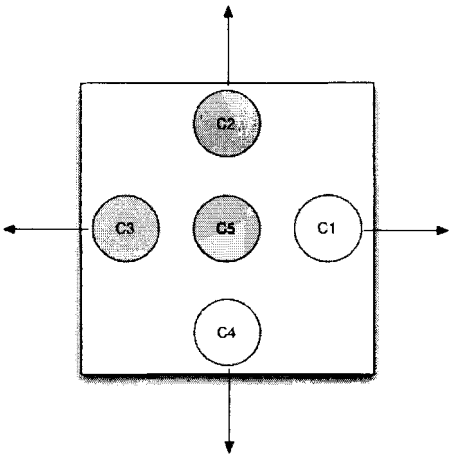

Figure 2.8: A 2D LGCA cell with 4 velocity channels and 1 rest channel. Filled dot represents the presence of a particle in the channel. Arrows represent the moving directions of a particle in that channel.

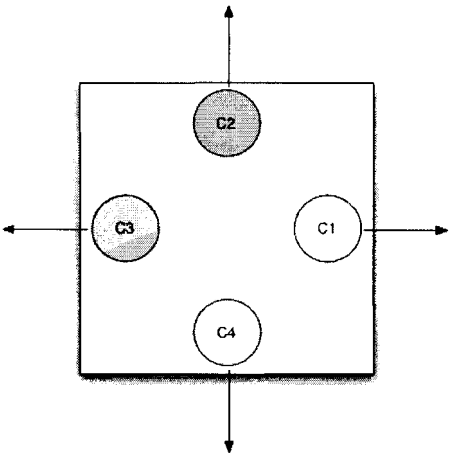

Figure 2.9: A 2D LGCA cell with 4 velocity channels and no rest channel. Filled dot represents the presence of a particle in the channel. Arrows represent the moving directions of a particle in that channel.

which can be decomposed into two basic operations: Propagation $\mathcal{P}$ and local Interaction $\mathcal{I}$

$$\mathcal{C} = \mathcal{P} \circ \mathcal{I}. \tag{2.3}$$

The dynamics of a LGCA arises from repetitive applications of superpositions of $\mathcal{I}$ and $\mathcal{P}$ applied simultaneously at all lattice cells at each discrete time step under the exclusion principle restriction.

During propagation, each particle moves from its channel to the corresponding channel of a neighbor cell in the direction of their velocity; i.e., a particle residing in channel (r, c_i) at time t jumps to channel $(r + mc_i, c_i)$ at time $t + 1$. The states of channels $(r + mc_i, c_i)$ after propagation are then given by

$$\eta_i(r + mc_i, t + 1) = \eta_i(r, t), i = 1, ...s \tag{2.4}$$

where $m \in \mathbb{N}$ is the speed of the particle.

Figure 2.10 shows an example of particles propagating in a one-dimensional LGCA with periodic boundary conditions. More information about the periodic boundary conditions for LGCA will be given in Chapter 4.

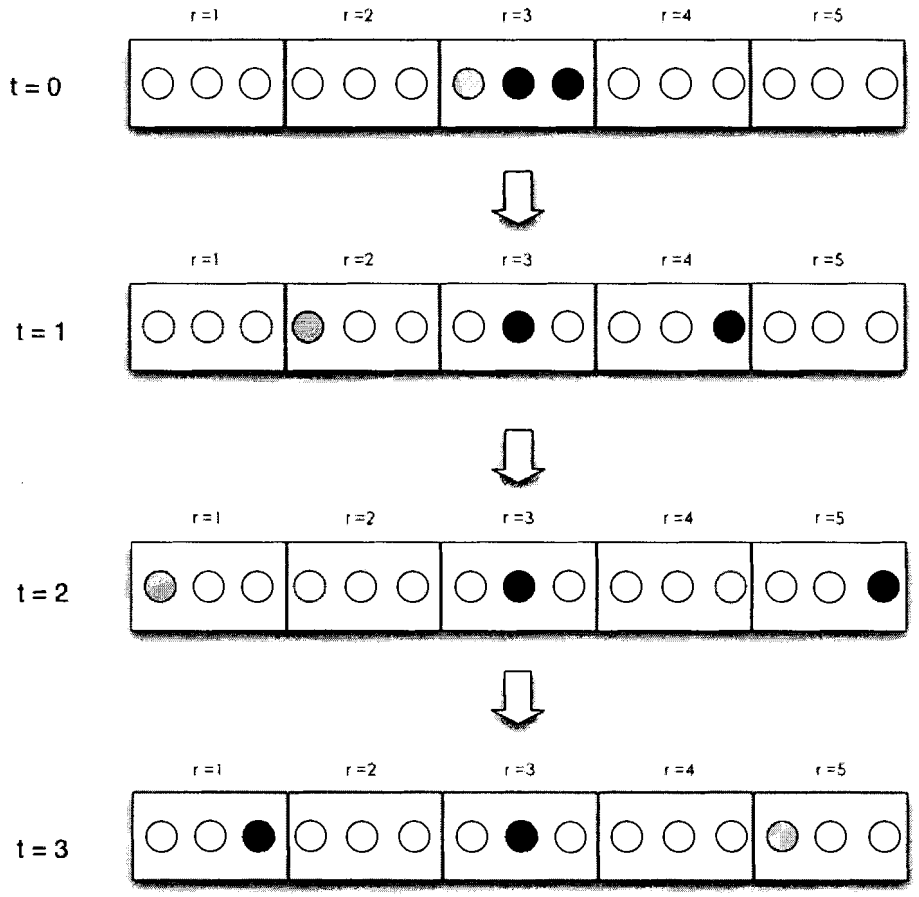

Figure 2.10: An example of the propagation process with speed $m = 1$ in a one-dimensional LGCA with five cells and two velocity channels and one rest channel on each cell. Filled dot represents the presence of a particle in the channel. 3 particles are labeled by different colors. Periodic boundary conditions are applied.

The state of cell r is also governed by the local Interaction operator $\mathcal{I}$ which determines the state of cell r based on the information from itself and its neighboring cells. The interaction neighborhood $\mathcal{N}(r)$, a set of cells defined for each cell $r \in \mathcal{L}$, is then introduced. The state of cell r is dependent of the states of the cells in $\mathcal{N}(r)$. Using $\eta_i^I(r,t)$ to denote the postinteraction state of cell r at time t, the state of channel $(r + mc_i, c_i)$ after local interaction and propagation is given by

$$\eta_i(r + mc_i, t + 1) = \eta_i^I(r,t), \qquad r \in \mathcal{L}, t \in \mathbb{N}, i = 1, ..., s. \tag{2.5}$$

2.2 Turing Pattern Formation

In [5], Turing pointed out diffusion plays a very important role in the loss of stability of a spatially homogeneous stable steady state of a reactive system. This type of instability is called *Turing instability*. A spatially heterogeneous pattern may arise in this situation, which is usually referred to as Turing pattern formation.

2.2.1 PDE Approach

Turing-type pattern (Diffusion-driven pattern) formation can be well captured and interpreted by macroscopic continuous models such as reaction-diffusion partial differential equations (PDEs). Consider a general one-dimensional system of reaction-diffusion PDEs,

$$\begin{cases} \frac{\partial u}{\partial t} = D_u \frac{\partial^2 u}{\partial x^2} + f(u,v) \\ \frac{\partial v}{\partial t} = D_v \frac{\partial^2 v}{\partial x^2} + g(u,v), \end{cases} \tag{2.6}$$

where $u = u(x,t)$, $v = v(x,t)$, $x \in [0, L]$, D_u, D_v are constants, and $f(u,v), g(u,v)$ usually are nonlinear functions obtained from reaction rate laws. The system can be written in the vector form,

$$\frac{\partial \vec{u}}{\partial t} = \vec{f}(\vec{u}) + D\frac{\partial \vec{u}^2}{\partial^2 x}, \tag{2.7}$$

with $D = \begin{bmatrix} D_u & 0 \\ 0 & D_v \end{bmatrix}$.

The spatially uniform steady state, denoted by $\vec{u_0} = (\bar{u}, \bar{v})$ can be obtained by solving

$\overrightarrow{f}(\overrightarrow{u}) = 0$, i.e., $\overrightarrow{f}(\overrightarrow{u_0}) = 0$. To obtain the conditions for Turing instability we linearize (2.7) around $\overrightarrow{u_0}$. Letting $u = \tilde{u} - \bar{u}$, $v = \tilde{v} - \bar{v}$, where $\tilde{u}, \tilde{v}$ are small, substituting into (2.7) and dropping high order terms, we then have

$$\frac{\partial \overrightarrow{w}}{\partial t} = J\overrightarrow{w} + D\frac{\partial \overrightarrow{w}^2}{\partial^2 x}, \tag{2.8}$$

where $\overrightarrow{w} = (\tilde{u}, \tilde{v})$ and $J = \begin{bmatrix} f_u & f_v \\ g_u & g_v \end{bmatrix}_{(\bar{u},\bar{v})}$.

Assuming the solution of the perturbed system (2.8) has the form of $\overrightarrow{w}(x,t) = \overrightarrow{W}e^{ikx}e^{\lambda t}$, then we have

$$\frac{\partial \overrightarrow{w}}{\partial t} = \lambda \overrightarrow{W} e^{ikx} e^{\lambda t}$$

and

$$\frac{\partial^2 \overrightarrow{w}}{\partial x^2} = -k^2 \overrightarrow{W} e^{ikx} e^{\lambda t}.$$

Substituting these into (2.8) and cancelling the common factor $e^{ikx}e^{\lambda t}$ gives

$$\lambda \overrightarrow{W} = (J - k^2 D)\overrightarrow{W},$$

i.e., λ, the growth rate, is an eigenvalue of $A = J - k^2 D = \begin{bmatrix} f_u - D_u k^2 & f_v \\ g_u & g_v - D_v k^2 \end{bmatrix}$, with

$$\lambda^2 - Tr(A)\lambda + det(A) = 0.$$

Solving for λ gives

$$\lambda_{1,2} = \frac{1}{2}(Tr(A) \pm \sqrt{\triangle}),$$

where $\triangle = Tr(A)^2 - 4det(A)$, $Tr(A) = f_u + g_v - k^2(D_u + D_v)$ and $det(A) = D_u D_v k^4 - (Dug_v + D_v g_u)K^2 + (f_u g_v - f_v g_u)$.

For Turing instability, the system should be linearly stable in the absence of diffusion ($k = 0$), which means that two eigenvalues λ_1, λ_2 are both negative. We have the first two conditions for Turing instability ,

$$Tr(A) = f_u + g_v < 0 \quad \text{and} \quad det(A) = f_u g_v - f_v g_u > 0.$$

For Turing instability (diffusion-driven instability), the spatially uniform stable steady state then loses its stability in the presence of diffusion which gives another condition for Turing instability,

$$det(A) = D_u D_v k^4 - (Dug_v + D_v g_u)K^2 + (f_u g_v - f_v g_u) < 0, \quad \text{for some} \quad k \neq 0.$$

Letting $h(k^2) = det(A) = D_u D_v k^4 - (D_u g_v + D_v f_u)k^2 + (f_u g_v - f_v g_u)$, since $(f_u g_v - f_v g_u) > 0$ (Turing condition 2) and $D_u D_v k^4 > 0$, the necessary condition for $h(k^2) < 0$ is $D_u g_v + D_v f_u > 0$ or $D_0 g_v + g_u > 0$ with $D_0 = \frac{D_v}{D_u}$.
Differentiating $h(k^2)$ with respect to k^2 and letting the derivative be zero yields $k_m^2 = \frac{D_u g_v + D_v f_u}{2D_u D_v}$. Substituting into $h(k^2)$ gives $h(k_m^2) = f_u g_v - f_v g_u - \frac{(D_u g_v + D_v f_u)^2}{4D_u D_v} < 0$, i.e.,

$$D_u g_v + D_v f_u > 2\sqrt{D_u D_v (f_u g_v - f_v g_u)} \quad \text{or} \quad g_v + D_0 f_u > 2\sqrt{D_0 (f_u g_v - f_v g_u)},$$

where $f_u g_v - f_v g_u > 0$ (Turing condition 2).
In summary, Turing conditions for Turing instability are

$$\text{Turing condition 1:} \quad f_u + g_v < 0, \tag{2.9}$$

$$\text{Turing condition 2:} \quad f_u g_v - f_v g_u > 0, \tag{2.10}$$

$$\text{Turing condition 3:} \quad D_0 f_u + g_v > 0, \tag{2.11}$$

$$\text{Turing condition 4:} \quad g_v + D_0 f_u > 2\sqrt{D_0 (f_u g_v - f_v g_u)}, \tag{2.12}$$

where $D_0 = \frac{D_v}{D_u}$.
If a reaction-diffusion system which has a spatially uniform steady state satisfies these four conditions, any spatially inhomogeneous perturbations to the spatially uniform steady state may lead to a spatially inhomogeneous steady solution which corresponds to a spatially inhomogeneous stationary pattern, i.e., Turing pattern formation. This analysis can be generalized to a two-dimensional reaction-diffusion system. We will investigate a two-dimensional reaction-diffusion system in Chapter 3. More detailed analysis of Turing pattern formation can be found in [3] and [4].

2.2.2 Activator-Inhibitor System

Multiplying (2.9) by (2.11) gives $D_0 f_u^2 + g_v^2 + (1 + D_0) f_u g_v < 0$, which implies that

$$f_u g_v < 0.$$

This means that f_u and g_v must have opposite signs. From (2.10) we have

$$f_v g_u < 0$$

since $f_u g_v < 0$, which means that f_v and g_u must have opposite signs, too. This leads to four possibilities for the Jacobian matrix $J = \begin{bmatrix} f_u & f_v \\ g_u & g_v \end{bmatrix}$:

$$\begin{bmatrix} + & - \\ + & - \end{bmatrix}, \begin{bmatrix} - & + \\ - & + \end{bmatrix}, \begin{bmatrix} + & + \\ - & - \end{bmatrix}, \begin{bmatrix} - & - \\ + & + \end{bmatrix}.$$

The first two matrices correspond to an activator-inhibitor system, where one species activates itself and the other, while the second species inhibits itself and the first. The last two correspond to an activator-substrate system, where one species activates itself and inhibits the second, while the second activates the first and inhibits itself. Assuming species u is the activator and species v the inhibitor, i.e., $f_u > 0$ and $g_v < 0$, then (2.9) and (2.11) become

$$f_u < -g_v = |g_v| \qquad \text{and} \qquad D > -\frac{g_v}{f_u} = \frac{|g_v|}{f_u},$$

which leads to

$$D = \frac{D_v}{D_u} > 1.$$

This means that for Turing instability the inhibitor should diffuse faster than the activator. An schematic representation for an activator-inhibitor system and an activator-substrate system are shown in Figure 2.11 and Figure 2.12, respectively.

2.2.3 CA Approach

The CA can also be used to model a reaction-diffusion system for the Turing pattern formation problem. In the framework of CA, the Turing pattern can be seen as a consequence of the local interactions between cells following certain rules. In contrast

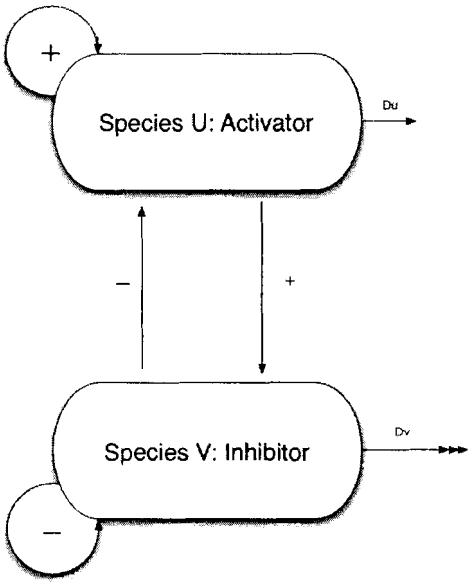

Figure 2.11: Graphical representation of the scheme of an activator-inhibitor system.

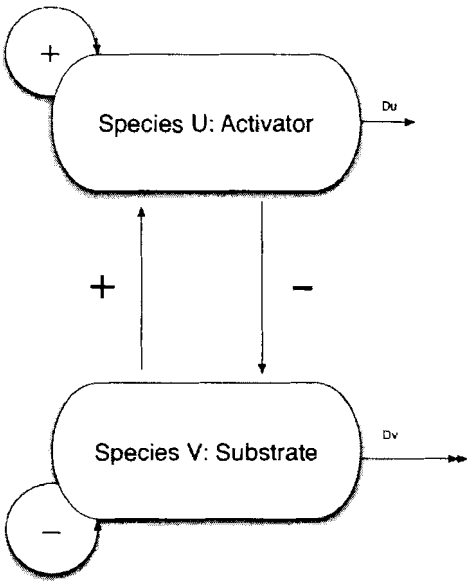

Figure 2.12: Graphical representation of the scheme of an activator-substrate system.

to the PDE approach, the CA model is totally discrete in time, space and state. Only a few states are used for each cell, and they are represented by integers. The CA model is not a replacement of the PDE model but an alternative way to investigate and analyze Turing pattern formation problems. In the next two chapters we will construct a CA model and a LGCA model for an activator-substrate system and an activator-inhibitor system, respectively. We will also compare the results with those obtained by solving a reaction-diffusion system using numerical methods in a qualitative way.

Chapter 3

Moving Average Cellular Automata

In this chapter, a cellular automata model for a reaction-diffusion system is introduced. We investigate 2D reaction-diffusion systems using so-called Moving Average CA first introduced in [8] for Turing-type pattern formation. The construction of the local CA rules can be obtained from a finite-difference-like discretization to the reaction-diffusion partial differential equations (PDEs), which relies on a moving-average procedure to implement the diffusive step and a probabilistic table lookup for the reactive step. A truncation rule is also introduced to force the operations to only involve integers. We apply this method to the two-dimensional Brusselator model. Turing-type patterns can emerge for well-chosen values of the parameters. The model is also simulated using the Second Order Backward Differentiation Formula for comparison.

3.1 Brusselator Model

The Brusselator model is an abstract model to demonstrate chemical oscillations and patterns. The reaction mechanism proposed by Prigogene and Lefever is

$$A \xrightarrow{k_1} X;$$

$$B + X \xrightarrow{k_2} Y + D;$$
$$2X + Y \xrightarrow{k_3} 3X;$$
$$X \xrightarrow{k_4} E;$$

where the $k_i's$ are the rate constants, and the reactant concentrations of A and B are kept constant. The governing ordinary differential equations for the concentrations of X and Y are

$$\begin{cases} \frac{d[X]}{dt} = k_1[A] - k_2[B][X] + k_3[X]^2[Y] - K_4[X] \\ \frac{d[Y]}{dt} = k_2[B][X] - k_3[X]^2[Y]. \end{cases} \tag{3.1}$$

In [6], the corresponding reaction-diffusion PDEs are written in nondimensional form as

$$\begin{cases} \frac{\partial u}{\partial t} = a - (b+1)u + & u^2v + D_u\nabla^2 u \\ \frac{\partial v}{\partial t} = bu - & u^2v + D_v\nabla^2 v. \end{cases} \tag{3.2}$$

3.1.1 Linear Stability Analysis

The spatially homogeneous steady state $(\bar{u}, \bar{v}) = (a, b/a)$ is obtained by solving the system

$$\begin{cases} f(u,v) = a - (b+1)u + & u^2v = 0 \\ g(u,v) = bu - & u^2v = 0. \end{cases} \tag{3.3}$$

To study the stability of this spatially homogeneous steady state , we linearize (3.2) around $(\bar{u}, \bar{v}) = (a, b/a)$, which gives

$$\frac{\partial \vec{w}}{\partial t} = J\vec{w} + D\nabla^2\vec{w} \tag{3.4}$$

where $\vec{w} = (\tilde{u}, \tilde{v})$, $\tilde{u} = u - \bar{u}$, $\tilde{v} = v - \bar{v}$, $J = \begin{bmatrix} f_u & f_v \\ g_u & g_v \end{bmatrix}_{(\bar{u},\bar{v})}$ and $D = \begin{bmatrix} D_u & 0 \\ 0 & D_v \end{bmatrix}$.
Assuming the solution of the perturbed system (3.4) has the form of $\vec{w}(\vec{x}, t) = \vec{W}e^{i\vec{k}\vec{x}}e^{\lambda t}$, where $\vec{x} = (x, y)$, $\vec{k} = (k_1, k_2)$, then we have

$$\frac{\partial \vec{w}}{\partial t} = \lambda\vec{W}e^{i\vec{k}\vec{x}}e^{\lambda t}$$

and

$$\nabla^2 \overrightarrow{w} = (\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2})\overrightarrow{w} = -(k_1^2 + k_2^2)\overrightarrow{W} e^{i \overrightarrow{k} \overrightarrow{x}} e^{\lambda t} = -k^2 \overrightarrow{W} e^{i \overrightarrow{k} \overrightarrow{x}} e^{\lambda t},$$

where k:=$|\overrightarrow{k}| = \sqrt{k_1^2 + k_2^2}$. Substituing these into (3.4) and cancelling the common factor $e^{i \overrightarrow{k} \overrightarrow{x}} e^{\lambda t}$ gives

$$\lambda \overrightarrow{W} = (J - k^2 D)\overrightarrow{W},$$

i.e., λ, the growth rate, is an eigenvalue of $A = J - k^2 D = \begin{bmatrix} f_u - D_u k^2 & f_v \\ g_u & g_v - D_v k^2 \end{bmatrix}$, with

$$\lambda^2 - Tr(A)\lambda + det(A) = 0.$$

Solving for λ gives

$$\lambda_{1,2} = \frac{1}{2}(Tr(A) \pm \sqrt{\triangle}),$$

where $\triangle = Tr(A)^2 - 4det(A)$, with

$$Tr(A) = b - 1 - a^2 - k^2(D_u + D_v) \quad \text{and} \quad det(A) = k^4 D_u D_v + k^2[(1-b)D_v + a^2 D_u] + a^2.$$

In the absence of diffusion, i.e., $k = 0$, since $det(A) = a^2 > 0$, the uniform state is linearly stable when $Tr(A) < 0$ or $b < 1 + a^2$. At $b = b_H = 1 + a^2$, we have $Tr(A) = 0, det(A) > 0$, which gives

$$Re(\lambda) = 0, Im(\lambda) = \pm ia.$$

Therefore, the Hopf bifurcation occurs at $k = 0, b = b_c^H$, and oscillation corresponding to the periodic solution in time exists for $b > b_H$. In the presence of diffusion, i.e., $k \neq 0$, an unstable mode exists if $det(A) < 0$ or $det(A) \geqslant 0, Tr(A) > 0$. For a Turing bifurcation, because $Tr(A) = b - 1 - a^2 - k^2(D_u + D_v) < 0$ if $b < b_c^H = 1 + a^2$ at the onset of instability, one eigenvalue should be zero and the other real and negative, which results in $det(A) = 0$. Letting $h(k^2) = det(A)$, finding its minimum gives

$$\min h(k^2) = a^2 - \frac{(bD_v - D_v - a^2 D_u)^2}{4D_u D_v} = 0.$$

Solving for b yields

$$b = b_c^T = (1 + a\sqrt{\frac{D_u}{D_v}})^2.$$

Therefore, a Turing bifurcation occurs at $k \neq 0, b = b_c^T$. In summary, linear stability analysis around the uniform steady state gives

$$\text{Hopf bifurcation:} \qquad b_c^H = 1 + a^2, \tag{3.5}$$

$$\text{Turing bifurcation:} \qquad b_c^T = \left(1 + a\sqrt{\frac{D_u}{D_v}}\right)^2. \tag{3.6}$$

Assuming $0 < \frac{D_u}{D_v} < 1$, letting $b_c^T = b_c^H$, and solving for a gives

$$a = a_c = \frac{2\sqrt{\frac{D_u}{D_v}}}{1 - \frac{D_u}{D_v}}.$$

So $b_c^H > b_c^T$ if $a > a_c$, and Turing instability appears first as b increases, while $b_c^H < b_c^T$ if $0 < a < a_c$, and Hopf instability appears first as b increases. For more detailed study of this stability analysis see [3] and [4]. Thus, the condition for Turing pattern formation is

$$b_c^T < b < b_c^H. \tag{3.7}$$

Figure 3.1 shows the bifurcation diagram of the Brusselator for $\frac{D_u}{D_v} = \frac{1}{3}$ in the $a - b$ plane. From it we can see that lines b_c^T, b_c^H and a_c divide the plane into 6 different regions, A, B, C, D, E, F. We will investigate the dynamics of the Brusselator by choosing different values of the parameters (a, b) in these 6 regions, numerically using the 2-SBDF method and moving average CA method. According the linear stability analysis, if start with (a, b) in region A, the system should be linearly stable. But as we increase the value of b with fixed a, Hopf bifurcation should appear first as b enters region B or C. Similarly, if we start with (a, b) in region D, the system is linearly stable. As we increase the value of b with fixed a, theTuring bifurcation should appear first and Turing patterns will be formed.

In the following sections we will first use 2-SBDF, an Implicit-Explicit finite difference method, to investigate the dynamics of Brusselator. Then, we will review the ideas of the Moving Average method for the reaction-diffusion system and use it to construct a CA model for the 2D Brusselator to investigate its dynamics for different

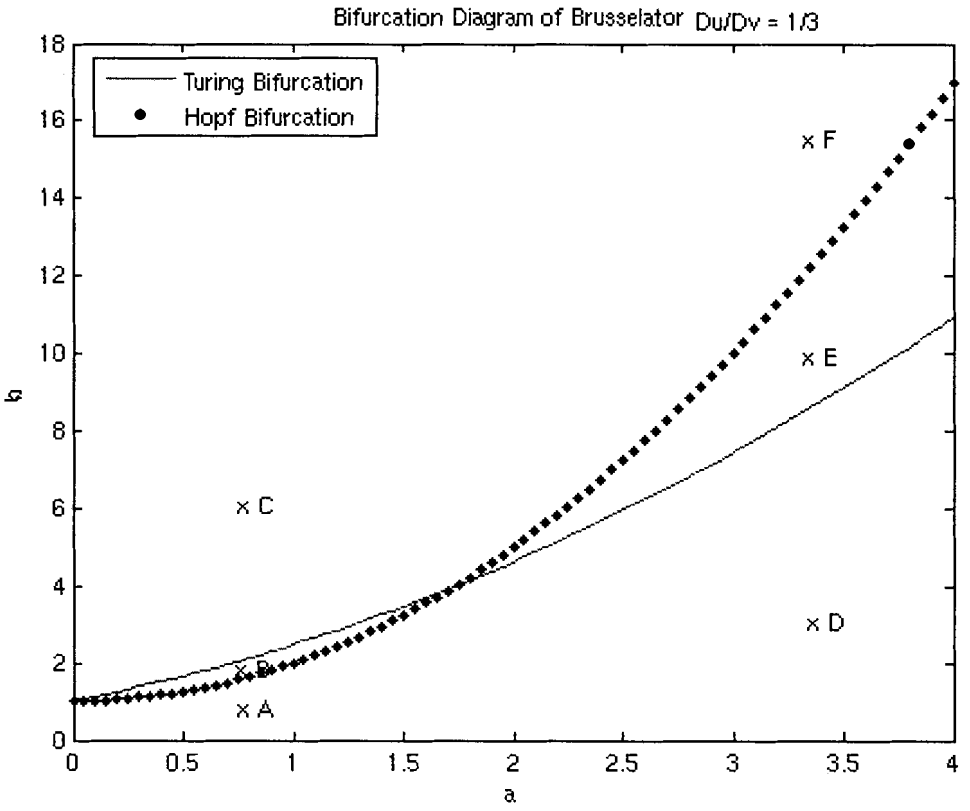

Figure 3.1: Bifurcation Diagram of Brusselator. The Turing-type pattern emerges when (a,b) lies in region that is above the solid line and below the dashed line.

values of the parameters. Some comparisons between these two methods are also given.

3.2 Numerical Simulation For Brusselator Using Finite Difference Method

In this section we study the numerical solution of system (3.2) using an Implicit-Explicit numerical scheme. This class of methods was systematically studied in [9]. Here system (3.2) is discretized by a Second Order Backwards Differentiation Formula (2-SBDF) by which the reaction terms are treated explicitly while the diffusion terms are treated implicitly [9], i.e.,

$$\frac{3W^{n+1} - 4W^n + W^{n-1}}{2\triangle t} = 2f(W^n) - f(W^{n-1}) + D\,\triangle_h W^{n+1}, \tag{3.8}$$

where $\triangle_h$ is the laplacian operator

$$\triangle_h W_{i,j} = \frac{W_{i-1,j} + W_{i,j-1} + W_{i,j+1} + W_{i+1,j} - 4W_{i,j}}{h^2}$$

for the interior points.

Applying (3.8) to (3.2) gives

$$\begin{cases} (3I - 2\triangle t D_u \triangle_h)U^{n+1} = 4U^n - U^{n-1} + 4\triangle t f(U^n, V^n) - 2\triangle t f(U^{n-1}, V^{n-1}) \\ (3I - 2\triangle t D_v \triangle_h)V^{n+1} = 4V^n - V^{n-1} + 4\triangle t g(U^n, V^n) - 2\triangle t g(U^{n-1}, V^{n-1}). \end{cases} \tag{3.9}$$

Linear systems (3.9) are solved by the matlab built-in PCG (Preconditioned Conjugate Gradient) method at each time step with Neumann boundary conditions. We choose $\triangle t = 0.0125, \triangle x = \triangle y = h = 0.02$ for all simulations, and the domain is discretized in a $100X100$ mesh. Several simulations are performed for different (a, b) chosen in region A to region F. For investigating the Hopf bifurcation, i.e., onset of temporal oscillations of a spatially uniform state, we plot the average of u over the domain in the $t-u$ plane instead of a contour of u in the $x-y$ plane. For investigating the Turing bifurcation, we plot the contour of u in the $x-y$ plane.

We first investigate Hopf instability. We start with $a = 1 < a_c$ and initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$. When b is chosen to be in region A, all perturbations die out and u goes back to the homogeneous steady state as shown in Figure 3.2.

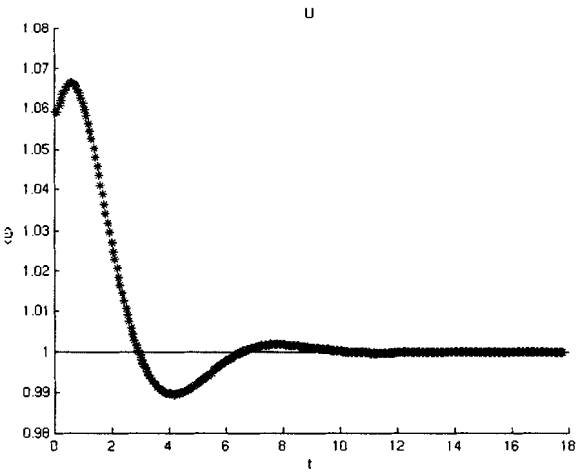

Figure 3.2: Evolution of u when $(a, b) = (1, 1)$ in region A with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$.

As we increase b with fixed a, a Hopf bifurcation occurs first. For example, when $b = a^2 + 1 + \epsilon$, $\epsilon = 0.01$, which means b is slightly above b_c^H and is in region B, we observe oscillations, and u is getting closer and closer to the asymptotic solution $u = a + \epsilon^{\frac{1}{2}} sin(\omega t)$ and the frequency ω is equal to a. If we keep increasing b to region C, we still observe oscillations as shown in Figure 3.4.

Then we investigate the Turing instability. We start with $a = 3 > a_c$ and the initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$. When b is chosen to be in regions D and E, $b = 4$ and $b = 8$ for instance, numerical simulations show that u goes back to the uniform steady state eventually since the system is linearly stable to homogeneous perturbations.

From linear theory we know that in regions E and F the system is linearly stable to spatially homogeneous perturbations but unstable to spatially inhomogeneous perturbations which gives rise to Turing patterns. Numerical simulations confirmed this. We choose $(a, b) = (3, 8)$ and $(a, b) = (3, 16)$. The initial condition is $u_0 = a + 0.1rand, v_0 = b/a + 0.1rand$, where $rand$ is a sequence of random numbers uniformly generated in $(0, 1)$ by the matlab. Stationary patterns emerge in this case

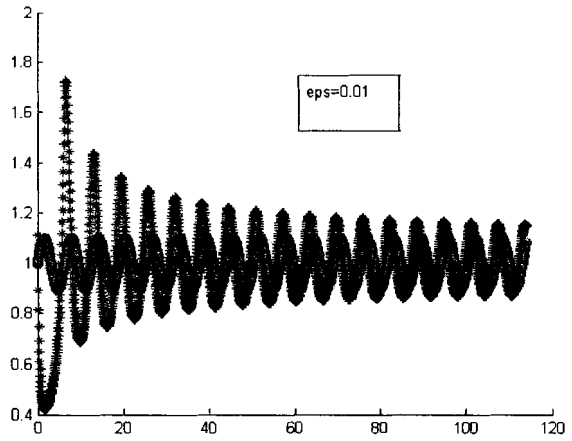

Figure 3.3: Evolution of u when $(a, b) = (1, 2 + \epsilon), \epsilon = 0.01$ in region B with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$. Hopf bifurcation occurs first. Numerical solution approaches asymptotic solution as time evolves.

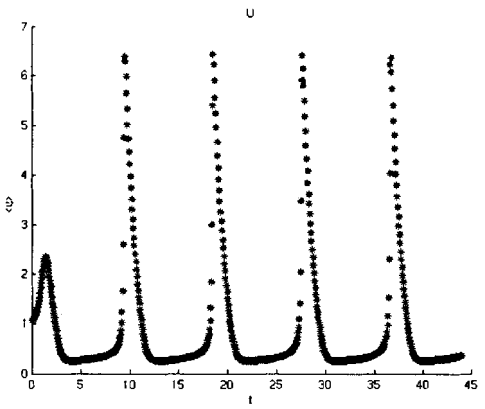

Figure 3.4: Evolution of u when $(a, b) = (1, 6)$ in region C with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$.

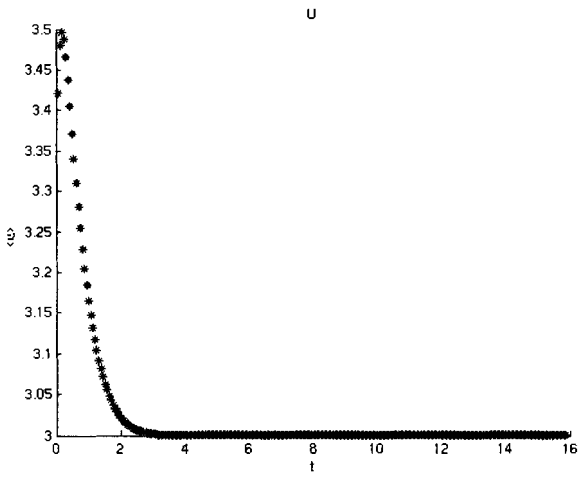

Figure 3.5: Evolution of u when $(a, b) = (3, 4)$ in region D with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$.

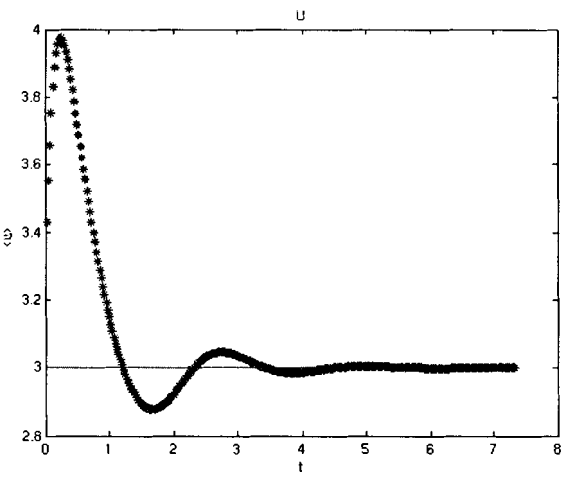

Figure 3.6: Evolution of u when $(a, b) = (3, 8)$ in region E with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$.

as shown in Figure 3.7 and Figure 3.8. Numerical simulations show that the Brusse-

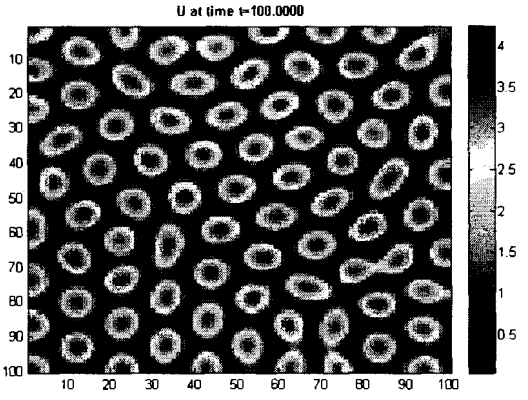

Figure 3.7: Concentration of u when $(a, b) = (3, 8)$ in region E with initial condition $u_0 = a+0.1rand, v_0 = b/a+0.1rand$.

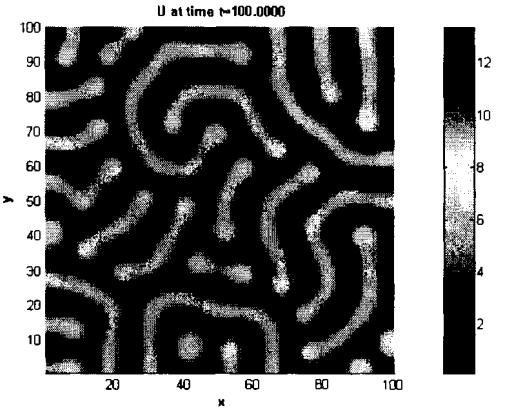

Figure 3.8: Concentration of u when $(a, b) = (3, 16)$ in region F with initial condition $u_0 = a+0.1rand, v_0 = b/a+0.1rand$.

lator is capable of generating Turing type patterns with the right choice of values of the parameters. In the next section we will give another type of discretization of the Brusselator model by cellular automata which also can capture the Turing bifurcation and Hopf bifurcation.

3.3 A MACA Model for 2D Brusselator

In this section we will investigate the 2D Brusselator model using the so-called Moving Average CA which was first introduced in [8]. The core part of CA modeling is the construction of CA updating rules. So, first of all, we will briefly review the ideas of deriving CA updating rules for Moving Average CA. Then we will utilize these rules to construct a CA model for the 2D Brusselator.

3.3.1 CA Updating Rules

Consider the system

$$\begin{cases} \frac{\partial u}{\partial t} = D_u \nabla^2 u + f(u,v) \\ \frac{\partial v}{\partial t} = D_v \nabla^2 v + g(u,v), \end{cases}$$

where $u = u(x,y,t)$, $v = v(x,y,t)$, D_u, D_v are constants, and $f(u,v), g(u,v)$ are nonlinear functions obtained from reaction rate laws. The complete dynamics of the CA are given by the repetitive application of three operators: $\mathcal{T}, \mathcal{R}, \mathcal{D}$ to each cell simultaneously.

Diffusion Rule $\mathcal{D}$

The CA rules for the diffusive part can be obtained from a modified finite difference scheme [8]. Let's start from the one-dimenstional diffusion equation

$$\frac{\partial u(x,t)}{\partial t} = D \frac{\partial^2 u(x,t)}{\partial x^2}. \tag{3.10}$$

Discretizing in time and space gives

$$\frac{u(x,t+\triangle t) - u(x,t)}{\triangle t} = D \frac{u(x+\triangle x,t) - 2u(x,t) + u(x-\triangle x,t)}{\triangle x^2}, \tag{3.11}$$

which can be rewritten as

$$u(x,t+\triangle t) = b_{-1}u(x-\triangle x,t) + b_0 u(x,t) + b_1 u(x+\triangle x,t) = \sum_{i=-1}^{1} b_i u(x+i\cdot\triangle x,t) \tag{3.12}$$

where $b_{-1} = b_1 = \lambda$, $b_0 = 1 - 2\lambda$, $\lambda = D\frac{\triangle t}{\triangle x^2}$.
Generalizing this discretization and using more general coefficients a_i yields,

$$u(x,t+\triangle t) = \sum_{i=-R}^{R} a_i \cdot u(x+i\cdot\triangle x,t). \tag{3.13}$$

Taylor expanding $u(x+i\cdot\triangle x,t)$ on the right-hand side in space gives,

$$u(x+i\cdot\triangle x,t) = u(x,t) + (i\triangle x)\cdot\frac{\partial u}{\partial x} + \frac{1}{2}(i\triangle x)^2\frac{\partial^2 u}{\partial x^2} + O(\triangle x^3). \tag{3.14}$$

Plugging this back into (3.13) produces

$$u(x,t+\triangle t)=\sum_{i=-R}^{R}a_i\cdot u(x,t)+\sum_{i=-R}^{R}(i\triangle x)a_i\cdot\frac{\partial u}{\partial x}+\frac{1}{2}\sum_{i=-R}^{R}(i\triangle x)^2a_i\cdot\frac{\partial^2 u}{\partial x^2}+O(\triangle x^3). \tag{3.15}$$

If we let

$$\begin{cases}\gamma_1:=\sum_{i=-R}^{R}a_i=1\\ \gamma_2:=\sum_{i=-R}^{R}(i\triangle x)a_i=0\\ \gamma_3:=\sum_{i=-R}^{R}(i\triangle x)^2a_i=2\triangle tD,\end{cases} \tag{3.16}$$

we obtain an approximation to the diffusion equation (3.10). Now we can define the Diffusive operator $\mathcal{D}$ as

$$\mathcal{D}(u(x,t))=u(x,t)+\frac{1}{2}\gamma_3\frac{\partial u^2}{\partial^2 x}. \tag{3.17}$$

A special case is obtained by making all a_i equal [8]. In this case, equation (3.13) becomes

$$u(x,t+\triangle t)=\sum_{i=-R}^{R}a_i\cdot u(x+i\cdot\triangle x,t)=a\cdot\sum_{i=-R}^{R}u(x+i\cdot\triangle x,t), \tag{3.18}$$

From equation (3.16) we have

$$\begin{cases}a=\frac{1}{2R+1}\\ D=\frac{R(R+1)}{6}\frac{\triangle x^2}{\triangle t}.\end{cases} \tag{3.19}$$

Notice that the diffusion coefficient D can be controlled by R. So for a multispecies problem, we just use different values of R (interaction neighborhood) to get different diffusion coefficients provided that $\triangle x, \triangle t$ are the same for all species. The advantage of this choice of coefficients is the calculation of the new value $u(x,t+\triangle t)$ for each r only involves one addition and one subtraction per cell instead of (2R+1) additions, i.e.,

$$u(x+\triangle x,t+\triangle t)=u(x,t+\triangle t)+a\cdot(u(x+(R+1)\triangle x,t)-u(x-R\triangle x,t)). \tag{3.20}$$

In 2D, $u = u(x, y, t)$, we can have similar results

$$u(x,y,t+\triangle t) = \sum_{i=-R}^{R}\sum_{j=-R}^{R} a_{i,j}u(x+i\triangle x, y+j\triangle y, t) = a\cdot\sum_{i=-R}^{R}\sum_{j=-R}^{R} u(x+i\triangle x, y+j\triangle y, t), \tag{3.21}$$

where

$$\begin{cases} a = (\frac{1}{2R+1})^2 \\ D = \frac{R(R+1)}{6}\frac{\triangle x^2}{\triangle t}. \end{cases} \tag{3.22}$$

So in order to compute the right-hand side of equation (3.21), the local average, we only need to calculate a moving sum in the x-direction first, then in the y-direction, and finally divide it by the normalization coefficient a, i.e.,

$$\begin{cases} \text{In x-direction:} & \tilde{u}(x+\triangle x, y, t) = \tilde{u}(x,y,t) + u(x+(R+1)\triangle x, y, t) - u(x-R\triangle x, y, t) \\ \text{In y-direction:} & \tilde{\tilde{u}}(x, y+\triangle y, t) = \tilde{\tilde{u}}(x,y,t) + \tilde{u}(x, y+(R+1)\triangle y, t) - \tilde{u}(x, y-R\triangle y, t) \\ \text{Normalization:} & u(x,y,t+\triangle t) = a\tilde{\tilde{u}}(x,y,t) \end{cases} \tag{3.23}$$

where $\tilde{u}(x, y, t)$ represents the horizontal local average of cell (x, y) in $(x - R\triangle x, x + R\triangle x)$ at time t and $\tilde{\tilde{u}}(x, y, t)$ represents the vertical local average of updated cell (x, y), i.e., the local average of cell (x, y) in the $(2R + 1)$x$(2R + 1)$ neighborhood. To make this updating process more clear, we give an example of calculating the local average of each cell in 2D for $R = 2$ with periodic boundary conditions using this so-called Moving average method.

R=2

Assume lattice $\mathcal{L}$ contains $(N + 1)$x$(N + 1)$ cells. The state of cell (i,j) is represented by $u(i, j)$, where $i, j = 0, 1, ..., N$. Since $R = 2$, the interaction neighborhood is an extended Moore neighborhood which means each cell has $(2R + 1)$x$(2R + 1)$= 25 neighbors. Instead of simply adding the values of all $(2R + 1)$x$(2R + 1)$=25 cells in the neighborhood for each cell ,we first calculate the horizontal average of each cell. We start from cell $(2, 0)$ as shown in Figure 3.9. The horizontal local average of cell $(2, 0)$, denoted as $\tilde{u}_{2,0}$, is computed which is $\tilde{u}_{2,0} = u_{0,0} + u_{1,0} + u_{2,0} + u_{3,0} + u_{4,0}$. Then, according to (3.23), we have $\tilde{u}_{3,0} = \tilde{u}_{2,0} + u_{5,0} - u_{0,0}$. So instead of doing $2R + 1 = 5$

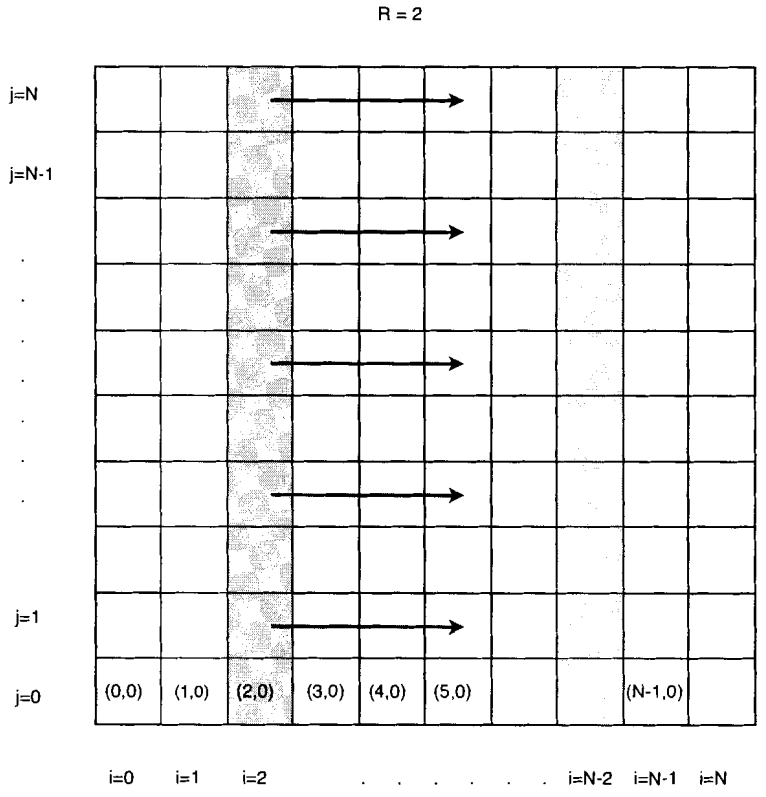

Figure 3.9: The local average is calculated in the horizontal direction.

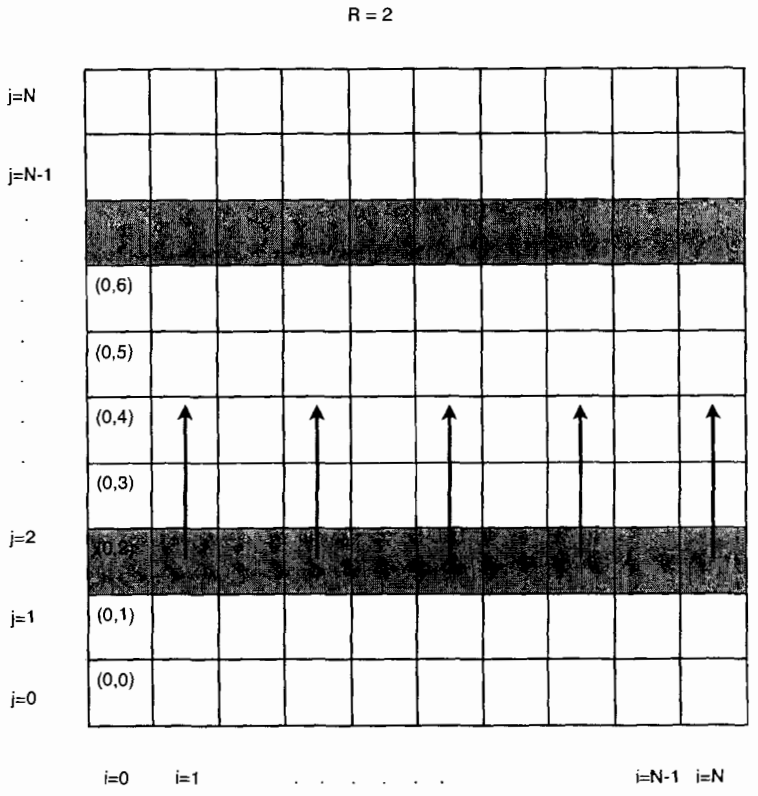

Figure 3.10: The local average is calculated in the vertical direction. Value of each cell has been updated by horizontal sum.

additions, it only involves two additions. Based on this algorithm, we can just "move" along the $x-$axis and calculate the horizontal sum of each interior cell as long as we know what the horizontal sum $\tilde{u}_{2,j}$ is for each $j = 1, 2, ..., N$. This algorithm can be applied to boundary cells $u_{i,j}$, $i = 0, 1, N-1, N, j = 0, ..., N$, if periodic boundary conditions, $u_{i,j} = u_{N+1+i,j}$ for $i = -1, -2$ and $j = 0, ..., N$ and $u_{i-1,j} = u_{N+i,j}$ for $i = 1, 2$ and $j = 0, ..., N$ are applied. Next, we will use the horizontal sum to calculate the vertical sum of each cell. Similarly, we first calculate the vertical sum of cell $(0, 2)$ as shown in Figure 3.10, which gives $\tilde{\tilde{u}}_{0,2} = \tilde{u}_{0,0} + \tilde{u}_{0,1} + \tilde{u}_{0,2} + \tilde{u}_{0,3} + \tilde{u}_{0,4}$. Notice that the value of each cell has been updated by the horizontal average in the previous step. Then, according to (3.23), we have $\tilde{\tilde{u}}_{0,3} = \tilde{\tilde{u}}_{0,2} + \tilde{u}_{0,5} - \tilde{u}_{0,0}$. Again, instead of doing $2R+1 = 5$ additions, it only involves two additions. Based on this, we can just "move" along the y-axis and calculate the vertical sum as long as we know what the vertical sum $\tilde{\tilde{u}}_{i,j}$ is for each $i = 1, ..., N$.. Boundary terms are treated similarly to the previous case. Figure 3.9 and Figure 3.10 give an illustration of these two updating processes. So we can see that the advantage of using this method is that we only need four additions per cell instead of 25 to calculate the local sum. The local average is then normalized by a to complete mimicing the diffusion process for one time step.

Reaction Rule

The Reaction operator also comes from a finite-difference-like discretization in time and space to the Reaction-Diffusion system [8] and is defined as

$$\mathcal{R}(u(x, y, t)) = u(x, y, t) + \triangle t f(u(x, y, t), v(x, y, t)). \tag{3.24}$$

Combining this with the diffusive operator $\mathcal{D}$ defined by (3.17), we have

$$\begin{aligned}
\mathcal{R}(\mathcal{D}(u(x, y, t))) &= \mathcal{R}(u(x, y, t) + \frac{1}{2}\gamma_3 \nabla^2 u(x, y, t)) \\
&= u(x, y, t) + \frac{1}{2}\gamma_3 \nabla^2 u(x, y, t) + \triangle t f(u(x, y, t), v(x, y, t)) + \gamma_3 \nabla^2 u(x, y, t)) \\
&= u(x, y, t) + \frac{1}{2}\gamma_3 \nabla^2 u(x, y, t) + \triangle t f(u(x, y, t), v(x, y, t)) + O(\triangle t^2).
\end{aligned} \tag{3.25}$$

Truncation Rule $\mathcal{T}$

Here we notice that the output of operator $\mathcal{R}$ might not be integers any more. Since the new state of each cell has to be represented by an integer, we need to introduce another operator to truncate the results. In general, the truncation rule is of two kinds: deterministic and probabilistic. The simplest case of a deterministic rule could be a rule that round the state of each cell to the nearest integer that is either greater than it or less than it. However, when testing this rule for the MACA model for the Brusselator, we have found that commonly it leads to the checkerboard instability for each cell. So we introduce a simple probabilistic rule which is defined as ([8])

$$\mathcal{T}(u) = \begin{cases} [u] & \text{with probability } 1-p \\ [u]+1 & \text{with probability } p \end{cases} \tag{3.26}$$

where $p = u - [u]$ and $[u]$ is the nearest integer that is less than u.
Two tables are needed for this rule, one giving $[u]$, the other giving p, for each possible output of the diffusion operator $\mathcal{D}$. The complete dynamics of the MACA is then given by $\mathcal{T} \circ \mathcal{R} \circ \mathcal{D}$.

3.3.2 Construction of MACA model for Brusselator

For modeling the 2D Brusselator by the MACA we need two two-dimensional lattices for species u and v, respectively. Each lattice contains the same number of cells. There is no channel at each cell, which differs from the lattice gas CA. The state of each cell is represented by an integer number in $[0, M]$, where M is the largest integer number each cell can have. So the state of each lattice can be represented by a two-dimentional matrix whose entries are only integer numbers in $[0, M]$. Different interaction neighborhoods are chosen for each species, which in turn gives different diffusion coefficients. Here we choose a Moore Neighborhood ($R_u = 1$) and an Extended Moore Neighborhood ($R_v = 2$). The CA updating rules are what we defined in the previous section. To make the updating process clearer, we give an algorithm for the MACA method:

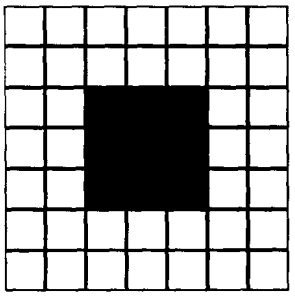

Figure 3.11: Moore Neighborhood

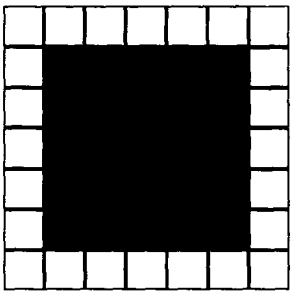

Figure 3.12: Extended Moore Neighborhood

1. Initialize u, v and set values of all parameters,
2. Calculate the horizontal sum using (3.23) for each cell and update each cell with the horizontal sum,
3. Calculate the vertical sum using (3.23) based on the updated values of each cell,
4. Rescale the value of each cell using (3.23),
5. Add the contribution from the reaction part,
6. Truncate the results using the proper rule to make sure each cell only contains an integer,
7. Update the values of all cells and go back to Step 2

Figure 3.13 shows an example of this updating process from time $t = 0$ to $t = \triangle t$ based on the above algorithm. For simplicity, $u_0 = 2$, $v_0 = 1$. We choose $\triangle t = 0.01$, $a = 3, b = 9$, $R_u = 1$ and $R_v = 2$, which means each cell of u and v has $(2R_u + 1)^2 = 9$ neighbors and $(2R_v + 1)^2 = 25$ neighbors, respectively. The horizontal sum is first calculated and then the vertical sum by which the values of all cells are updated. Next, the values of all cells of u, v are rescaled by $a_u = \frac{1}{2R_u+1}^2 = \frac{1}{9}$ and $a_v = \frac{1}{2R_v+1}^2 = \frac{1}{25}$, respectively. The contribution from reaction terms are then computed for u and v, which gives $\triangle t f(u, v) = -0.13$ and $\triangle t g(u, v) = 0.14$. Last, the values of all cells are truncated using probabilistic rule (3.26) and look-up tables.

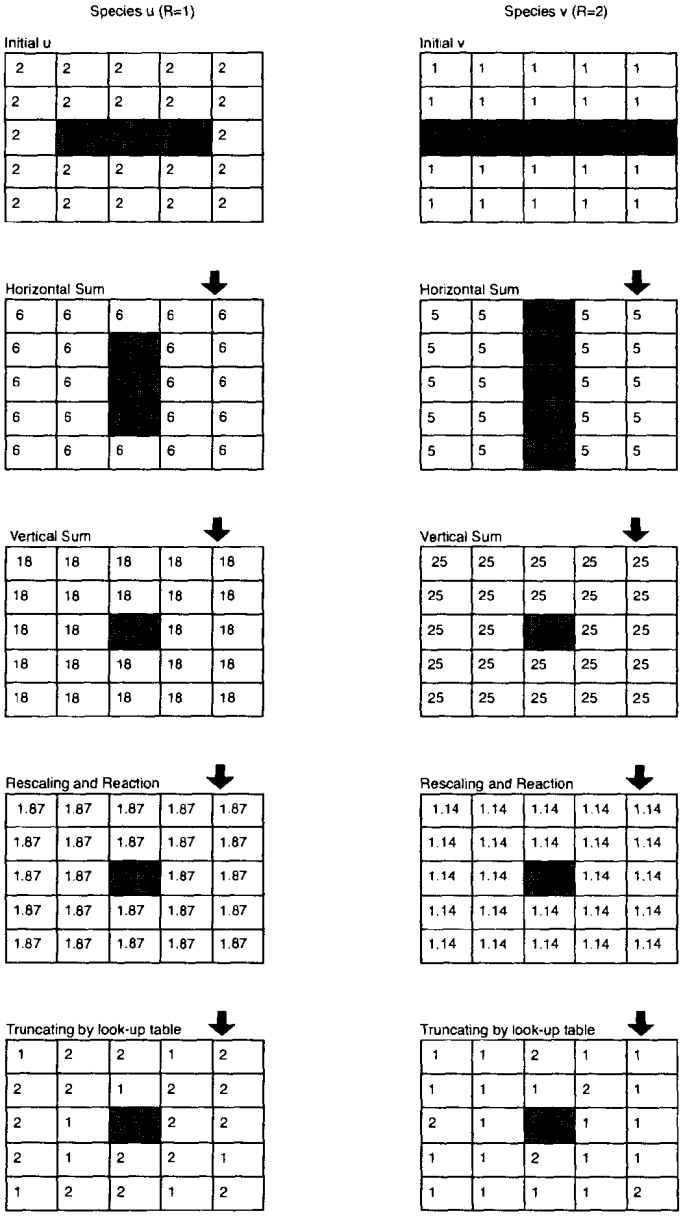

Figure 3.13: An example of MACA updating process from $t = 0$ to $t = \triangle t$.

3.3.3 Numerical Simulations using MACA

In this section several numerical simulations for the Brusselator using the MACA method are performed and the results are compared with those obtained by the 2-SBDF method in a qualitative way. We choose $R_u = 1, R_v = 2$ for species u and v, respectively, which gives us $\frac{D_u}{D_v} = \frac{1}{3}$. The initial conditions is $u_0 = a + 0.1a, v_0 = \frac{b}{a} + 0.1\frac{b}{a}$ or $u_0 = a + 0.1rand, v_0 = \frac{b}{a} + 0.1rand$, and periodic boundary conditions are applied. The Length of the lattice is $L = 250$ and contains NxN cells ($N = 250$), i.e., the spatial domain is discretized in a 250x250 mesh where $\triangle x = \triangle y = 1$ and time step $\triangle t = 0.01$. The discretization level is chosen to be $M_u = M_v = 10$, which means the CA model we construct only has 11 states for each cell. Concentrations of species u are plotted for different choices of parameters. Similarly, we plot the average of u over the domain in the $u - t$ plane and the contour plot of u in the $x - y$ plane to show the evolutions of u .

Firstly, we investigate the Hopf instability of the system by choosing $a < a_c$ and increasing b with fixed a. The initial condition is $u_0 = a + 0.1a, v_0 = \frac{b}{a} + 0.1\frac{b}{a}$. By the linear analysis a Hopf instability should appear as b passes b_c^H. We choose $(a, b) = (1, 1), (a, b) = (1, 3)$ and $(a, b) = (1, 9)$ in regions A, B and C of Figure 3.1, respectively. The average of u is plotted for each choice of parameters. Numerical simulations show that all perturbations vanished and u goes back to the spatially steady state $a = 1$ for all three examples. The $(a, b) = (1, 1)$ case confirms the linear stability analysis since the system is linearly stable to small perturbations. However, a Hopf instability does not occur as b passes b_c^H, the Hopf bifurcation line, since no oscillations are observed in Figure 3.15 and Figure 3.16. To investigate Hopf instability further, we change the value of a to be $a = 1.1$ and choose $(a, b) = (1.1, 5)$ and $(a, b) = (1.1, 9)$. We do not see Hopf instability immediately as b passes b_c^H, the Hopf bifurcation line. However, as we keep increasing b till it passes b_c^T, the Turing bifurcation line, a Hopf instability occurs first as shown in Figure 3.18 and Figure 3.19.

Secondly, we investigate the Turing instability. We choose $(a, b) = (3, 4), (a, b) =$

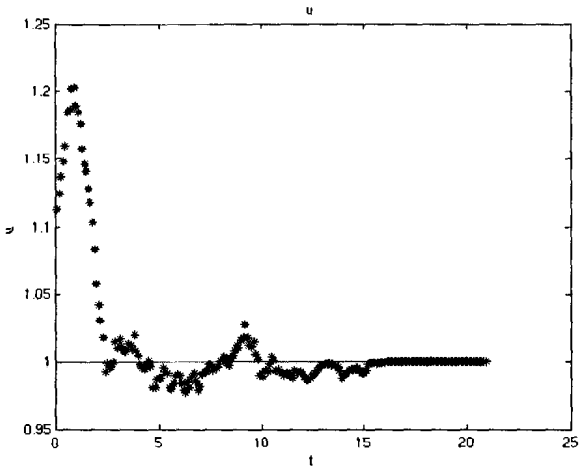

Figure 3.14: Evolution of average of u when $(a, b) = (1, 1)$ in region A with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$

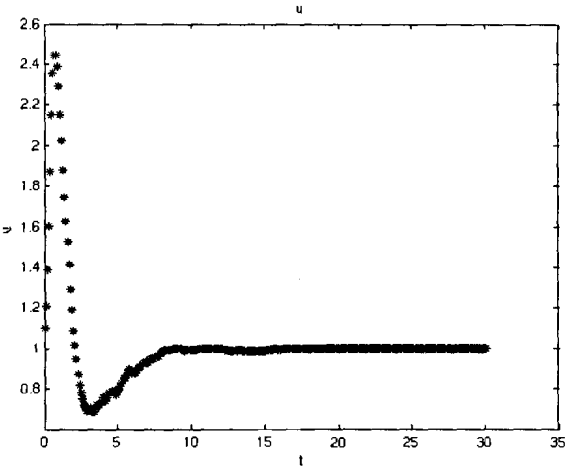

Figure 3.15: Evolution of average of u when $(a, b) = (1, 3)$ in region B with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$

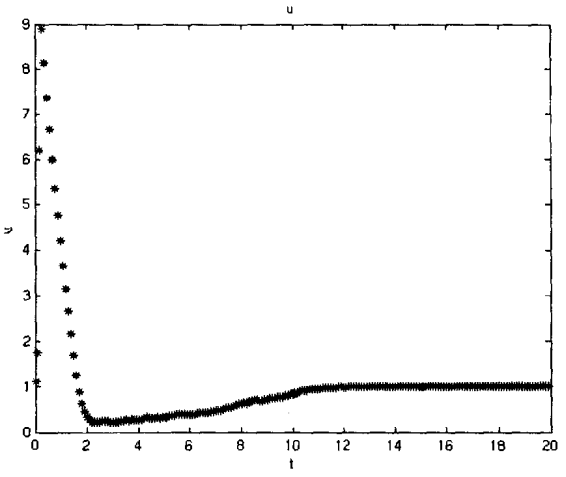

Figure 3.16: Evolution of average of u when $(a, b) = (1, 9)$ in region C with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$

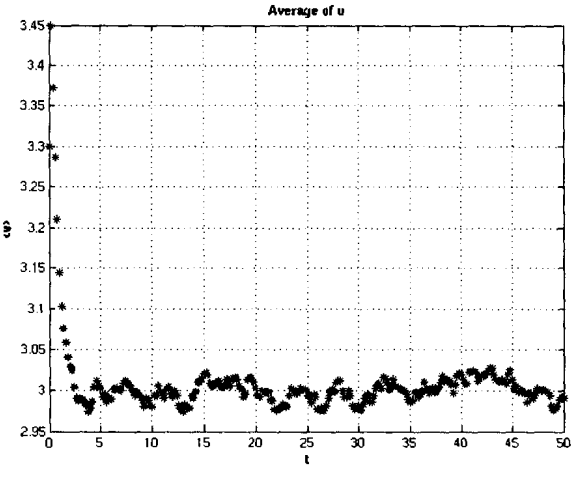

Figure 3.17: Evolution of average of u when $(a, b) = (3, 4)$ in region D with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$

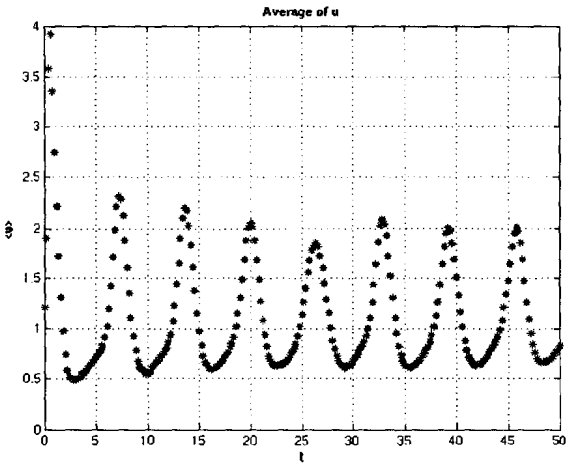

Figure 3.18: Evolution of average of u when $(a, b) = (1.1, 5)$ in region C with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$

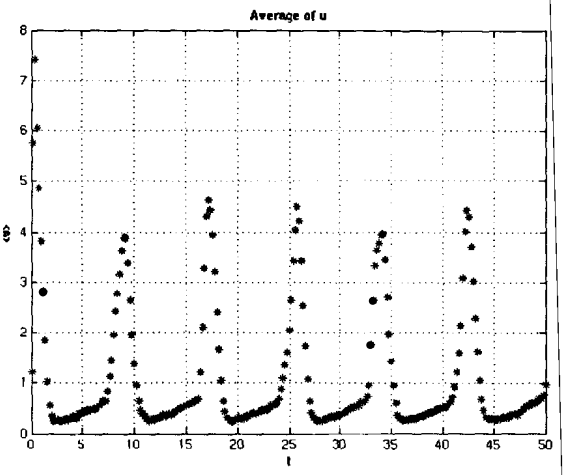

Figure 3.19: Evolution of average of u when $(a, b) = (1.1, 9)$ in region C with initial condition $u_0 = a + 0.1a, v_0 = b/a + 0.1b/a$

$(3, 9)$ and $(a, b) = (3, 12), (a, b) = (3, 18)$ in regions D, E and F of Figure 3.1, respectively. When $(a, b) = (3, 4)$, Figure 3.17 shows that starting with $u_0 = a + 0.1a, v_0 = \frac{b}{a} + 0.1\frac{b}{a}$, u is going back to the spatially steady state $a = 3$ with small oscillations (± 0.05). Starting with $u_0 = a + 0.1rand, v_0 = \frac{b}{a} + 0.1rand$, we observe that the Turing instability occusr and stationary patterns are generated as b passed b_c^T, the Turing bifurcation line, as shown in Figure 3.20, Figure 3.21 and Figure 3.22, which confirms the linear stability analysis.

Lastly, we also find that Turing patterns are observed when (a, b) is in region C. We choose $(a, b) = (1.5, 9)$. The initial condition is $u_0 = a + 0.1rand, v_0 = \frac{b}{a} + 0.1rand$. Isolated spot-like patterns are generated as shown in Figure 3.23.

From the above simulations we see that the MACA model is capable of capturing the Turing and Hopf instability with well-chosen values of the parameters a and b, and it gives the qualitatively the same results as those obtained by the 2-SBDF method. To our knowledge, this is the first numerical study of the Brusselator based on the MACA method and corresponding comparisons. All of the simulations are computed in Matlab, and the matlab codes for the MACA model can be found in the Appendix.

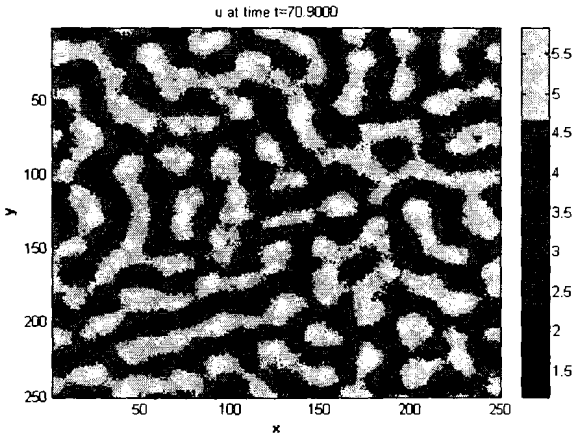

Figure 3.20: Concentration of u when $(a, b) = (3, 9)$ in region E with initial condition $u_0 = a+0.1rand, v_0 = b/a+0.1rand$.

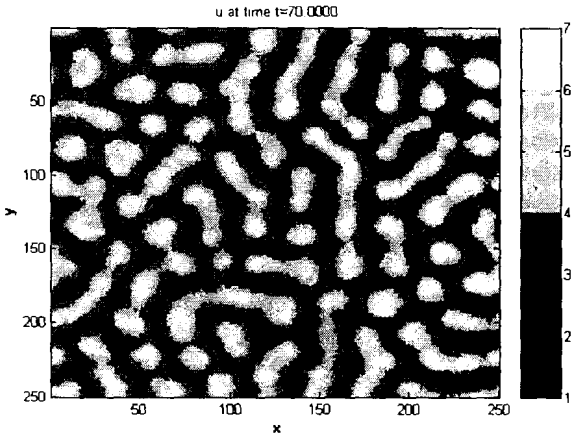

Figure 3.21: Concentration of u when $(a, b) = (3, 12)$ in region F with initial condition $u_0 = a+0.1rand, v_0 = b/a+0.1rand$.

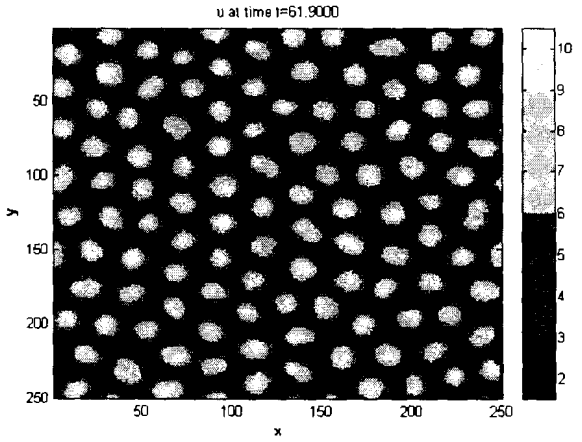

Figure 3.22: Concentration of u when $(a, b) = (3, 18)$ in region F with initial condition $u_0 = a+0.1rand, v_0 = b/a+0.1rand$.

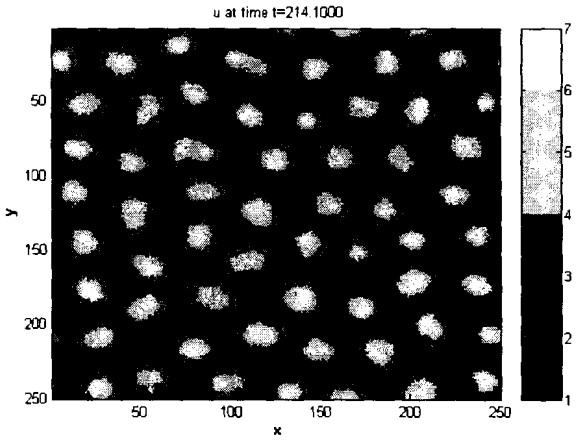

Figure 3.23: Concentration of u when $(a, b) = (1.5, 9)$ in region C with initial condition $u_0 = a+0.1rand, v_0 = b/a+0.1rand$.

Chapter 4

LGCA for Turing Pattern Formation

In the previous chapter, we introduced a MACA model for the 2D Brussellator reaction-diffusion system. We started with the reaction-diffusion system, and utilized a MACA to discretize the system. In some sense, the MACA method can be seen as a special kind of numerical method for reaction-diffusion PDEs which only involves integer operations. In this chapter, we use another type of CA to model and analyze pattern formation problems: the Lattice Gas CA (LGCA). The LGCA has been widely used to model and simulate problems in hydrodynamics and in reaction-diffusion systems ([7][10][1]). From Chapter 2 we have seen the key difference between classical CA and LGCA is the introduction of channels. In the LGCA system, each cell contains several channels, on which particles residing will propagate following the Propagation rule $\mathcal{P}$. It is found and demonstrated that with the help of a particular random permutation of the channel occupation (Shuffling rule $\mathcal{M}$) at each cell, each particle performs a random walk and the macroscopic behavior of the particle density is expected to be diffusive (see [7] [10]). For modeling reaction-diffusion systems, Reaction rule $\mathcal{R}$ is constructed and combined with the Propagation rule and Shuffling rule to give the complete dynamics. Reaction rule $\mathcal{R}$ can be designed by using the knowledge of the chemical reaction equations (see for instance [7]) or using the knowledge of the mechanisms and properties of the system that is to be modeled (see for

instance [1]). From the definition of the LGCA system mentioned in Chapter 2, we know that the dynamics of a LGCA system also depends on three other things: lattice $\mathcal{L}$, state $\mathcal{E}$ and interaction neighborhood $\mathcal{N}$ by which the system is characterized.

In this chapter, firstly, we design a reaction rule based on the rule described in [1] and construct a LGCA model for an activator-inhibitor system in one- and two-dimensions using this modified reaction rule. This modified reaction rule incorporates the rule defined in [1] and is capable of showing more different and interesting patterns. An algorithm for constructing such a model is also presented. Secondly, using this rule, we investigate different dynamics and patterns in this LGCA model in one- and two-dimensions with different $\mathcal{L}$, $\mathcal{E}$ and $\mathcal{N}$. The role of the rest channel in the LGCA system is also investigated through numerical simulations. Comparisons between the LGCA model with a rest channel and without a rest channel in 1D and 2D are given, from which we shall see that the absence of the rest channel may cause the checkerboard instability, and that such instability could be eliminated by extending $\mathcal{N}$. Comparisons between different $\mathcal{N}$ are also given. Through these simulations and analysis we shall see how this LGCA model is characterized by the lattice $\mathcal{L}$, state $\mathcal{E}$ and interaction neighborhood $\mathcal{N}$. Lastly, we will study the commutativity of rules $\mathcal{P}$, $\mathcal{M}$ and $\mathcal{R}$ through the numerical simulations. To our knowledge, this is the first study of such models based on the modified reaction rule with corresponding comparisons.

4.1 Construction of LGCA Model for R-D System

In order to mimic the reaction-diffusion process of the pattern formation problem in the framework of LGCA, the construction of updating rules for the reaction and diffusion process is the key part. In Section 2.2.2, we know that the evolution of the LGCA system is determined by $\mathcal{C}$ that is the composite of Propagation rule $\mathcal{P}$ and Interaction rule $\mathcal{I}$, i.e.,

$$[state]^{t+1} = \mathcal{C}[state]^t = (\mathcal{P} \circ \mathcal{I})[state]^t.$$

For modeling a reaction-diffusion process, the Interaction rule $\mathcal{I}$ is decomposed into two rules, Shuffling rule $\mathcal{M}$ and Reaction rule $\mathcal{R}$. To mimic the diffusion process, Shuffling rule $\mathcal{M}$ is introduced and combined with Propagation rule $\mathcal{P}$. Reaction rule $\mathcal{R}$ can be designed by using chemical reaction equations of a particular chemical reaction (see [7]) or using the knowledge of the mechanism and properties of the system that is to be modeled (see [1]). Then the dynamics of the LGCA system can be described by

$$[state]^{t+1} = \mathcal{C}[state]^t,$$

where $\mathcal{C}$ consists of 3 steps: Propagation $\mathcal{P}$, Shuffling $\mathcal{M}$ and Reaction $\mathcal{R}$

$$\mathcal{C} = \mathcal{P} \circ \mathcal{M} \circ \mathcal{R}.$$

In this chapter, our simulations and discussions are based on the order $\mathcal{P} \circ \mathcal{M} \circ \mathcal{R}$. In fact, based upon our simulations, the order of these operators does not qualitatively affect the long time behaviors of the LGCA system. We will investigate this in the last section.

4.1.1 Random Walk in LGCA System

In this section we will numerically show that in a non-reactive 1D LGCA, $\mathcal{R}$ = identity, each particle executes a discrete random walk through the repeated application of $\mathcal{M}$ and $\mathcal{P}$, where $\mathcal{M}$ acts as follows ([1]):

1. Before the propagation step at each cell, each particle randomly and independently selects a new velocity (including the velocity it has now and zero velocity if a rest channel is present) among the values permitted by the lattice.

2. No more than one particle at each cell can select the same velocity or rest channel (Exclusion principle).

Consider the LGCA system that has s channels on each cell. At each time step, before propagating, particles will interchange their velocities randomly and independently at each cell, where independently means this process ignores the state and location of

each cell. There are $s!$ ways of shuffling at each cell. In order to realize this process, a sequence of Boolean random variables

$$\{\xi_1(r,t), ..., \xi_{s!}(r,t)\}, \qquad \forall r \in \mathcal{L}, t \in \mathbb{N},$$

is generated in order to select a permutation matrix. Notice that for every t the random variables are independent of the past evolution of the automaton and at a given node r and time step t only one of the ξ_j is equal to 1, which means only one permutation is selected at each cell r and each time step t, i.e.,

$$\sum_{j=1}^{s!} \xi_j(r,t) = \sum_{j=1}^{s!} P(\xi_j(r,t) = 1) = 1, \qquad \forall r \in \mathcal{L}, t \in \mathbb{N}. \tag{4.1}$$

The state of channel (r, c_i) at time t after the shuffling process, denoted by $\eta_i^{\mathcal{M}}(r,t)$, is then given by

$$\eta_i^{\mathcal{M}}(r,t) = \sum_{j=1}^{s!} \xi_j(r,t) \sum_{l=1}^{s!} \eta_l(r,t) a_{li}^j, \qquad i = 1, ..., s. \tag{4.2}$$

Here a_{li}^j is a matrix element of A_j, and $A_j \in \mathcal{A}_s$, where $\mathcal{A}_s$ is the set of all s by s orthonomal permutation matrices that is defined as

$$\mathcal{A}_s = \left\{A \in \mathbb{R}^{s\times s} : \exists \pi \in \Pi_s \mathbf{a}^i = \mathbf{u}^{\pi(i)}, \forall i = 1, ..., s\right\} = \{A_1, ..., A_{s!}\}$$

where $\mathbf{u}^j$ are unit vectors with s components, $j = 1, ..., s$, $\mathbf{a}^j$ is the ith column vector of A, and Π_s is the set of all permutations of s elements.
Combining with propagation step (2.4), the complete dynamics of the non-reactive LGCA system can be described by

$$\eta_i(r + mc_i, t+1) = \eta_i^{\mathcal{M}}(r,t), \qquad i = 1, ..., s. \tag{4.3}$$

Using (4.2) and (4.3) we can easily get the new state of each channel on lattice $\mathcal{L}$. Next, we will numerically show that a 1D LGCA system whose dynamics are determined by the above Propagation rule and Shuffling rule can mimic the random walk, and the macroscopic behavior can be expected to be diffusive.

Let's consider a 1D LGCA system with $N = 101$ cells and each cell has $s = 3$ channels (including 2 velocity channels and 1 rest channel). Initially, we have a particle residing on channel $(0, c_3)$ which is the rest channel at $r = 0$. At $Nt = 0$, this particle will randomly choose a velocity channel or stay where it is. From $Nt = 0$ to $Nt = 1$, it will jump to left or right or stay depending on the channel it chose at $Nt = 0$ with the same probability p=$\frac{1}{3}$. To investigate this process further, we do several simulations using Matlab with this initial condition. Figures 4.1 and 4.2 give four random walks with $Nt = 100$ time steps for different m, the speed of the particle. Statistical analysis is also given in order to show this process is random. We first calculate the average straight-line distance between start and finish point at different time step Nt for $Niter = 1000$, where $Niter$ is the number of simulations. Figures 4.3 and 4.4 show that the average straight-line distance between start and finish points of Nt steps is on the order of $\sqrt{Nt}$ and behaves like the theoretical value $m\sqrt{\frac{2}{3}\frac{2}{\pi}}\sqrt{Nt}$, where $m = 1, 2$.

Another thing we observe is the Gaussian structure in this LGCA system. We start

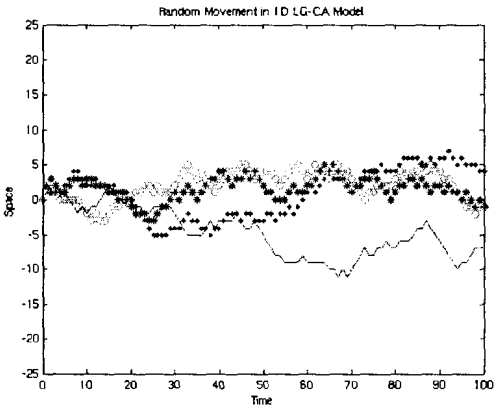

Figure 4.1: Four random walks starting at $r = 0$ in 1D LGCA system with $m = 1, Nt = 100, N = 101$.

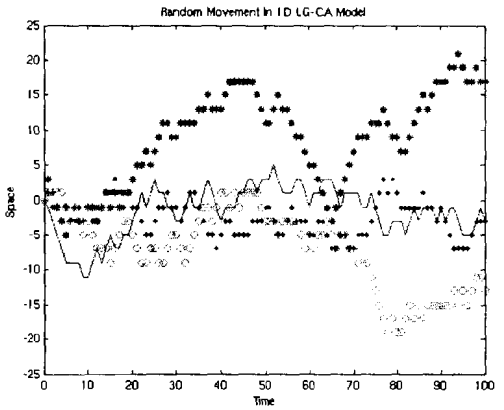

Figure 4.2: Four random walks starting at $r = 0$ in 1D LGCA system with $m = 2, Nt = 100, N = 101$.

with the single particle at channel $(0, c_3)$, and for each $Nt = 5, 25, 50$, $Niter = 10000$ simulations are performed with $m = 1$ and $N = 101$. The probability density function $p(r, t)$ of the particle is then computed, where $p(r, t) := P(n(r, t) = 1)$ represents the probability of the appearance of the particle at cell r at time t. Figure 4.5 shows the

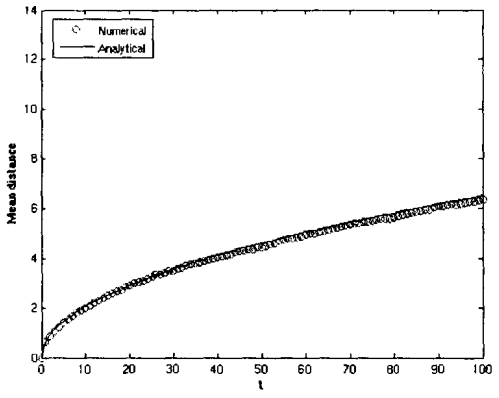

Figure 4.3: Mean distance between start and finish point with $m = 1$ for different time steps.

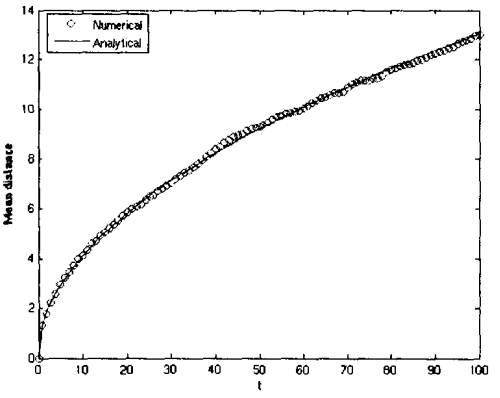

Figure 4.4: Mean distance between start and finish point with $m = 2$ for different time steps.

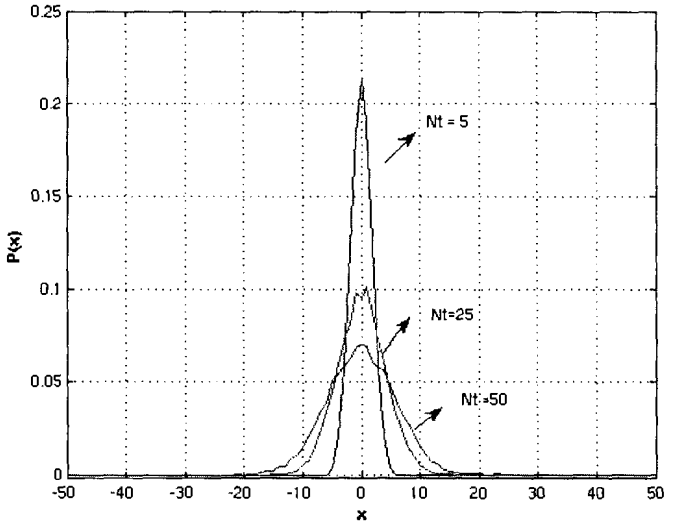

Figure 4.5: Gaussian Structure in LGCA system with $N = 101, Nt = [5, 25, 50], Niter = 10000$.

probability density functions for different time step $t = Nt$.

Equation (4.3) models a random walk for each particle on lattice $\mathcal{L}$. Since the interaction between random walkers is strictly local, the macroscopic behavior of the system can be expected to be *diffusive* ([7][10]).

4.1.2 Multicomponent LGCA System

For modeling an activator-inhibitor system, there are two species of interest involved: activator A and inhibitor I. Similar to the MACA model for the Brusselator, we need a lattice $\mathcal{L}_A$ and a lattice $\mathcal{L}_I$ for species A and I, respectively. For simplicity, we assume $\mathcal{L}_A$ and $\mathcal{L}_I$ are identical, i.e., they have the same number of cells and channels. Particles of species σ can only reside in the channels of species σ and propagate with the speeds m_σ, where $\sigma = \{A, I\}$. Thus, based on the Shuffling rule $\mathcal{M}$, each particle of species σ performs a random walk on the lattice $\mathcal{L}_\sigma$ independently. Since the propagation and shuffling process of one species are independent of those of the other species, the dynamics of a multicomponent LGCA system arises from the interactions between particles of the two species during the reaction process. The creation or destruction of particles at each cell of one species is determined by not only itself but the particles of the other species. Thus, the dynamics of a two-component LGCA system can be described by

$$\eta_{\sigma,i}(r + m_\sigma c_i, t + 1) = (\eta^{\mathcal{R}}_{\sigma,i}(r, t))^{\mathcal{M}}, \qquad \sigma = \{A, I\} \tag{4.4}$$

where $\eta^{\mathcal{R}}_{\sigma,i}(r, t)$ represents the state of channel (r, c_i) of species σ at time t after the reaction and $(\eta^{\mathcal{R}}_{\sigma,i}(r, t))^{\mathcal{M}}$ represents the state of channel (r, c_i) of species σ after the reaction and shuffling.

4.1.3 Reaction Rule for A-I System

The reaction rule in the LGCA system determines the construction or destruction of a particle in each channel on a lattice. The interaction neighborhood plays a very important role here since for a multicomponent LGCA system, the construction or destruction of a particle of one species in a channel may depend on the particles at

the same cell and the particles at the neighboring cells of the other species. Different dynamics arise in the LGCA system with different interaction neighborhoods. Here we present a modified reaction rule for the Activator-Inhibitor system based on the rule for the Activator-Inhibitor system described in [1]. We have

$$n_A^{\mathcal{R}}(r,t) = \begin{cases} s & \text{with probability } p_1 \quad \text{if} \quad D(r,t) > 0 \\ 0 & \text{with probability } p_2 \quad \text{if} \quad D(r,t) < 0 \\ n_A(r,t) & \text{otherwise} \end{cases} \tag{4.5}$$

and

$$n_I^{\mathcal{R}}(r,t) = \begin{cases} s & \text{with probability } p_3 \quad \text{if} \quad D(r,t) > 0 \\ 0 & \text{with probability } p_4 \quad \text{if} \quad D(r,t) < 0 \\ n_I(r,t) & \text{otherwise} \end{cases} \tag{4.6}$$

where $n_\sigma^{\mathcal{R}}(r,t)$ represents the total number of particles of species σ at cell r and time t after reaction, s is the number of channels each cell possesses, $D(r,t) = n_A(\mathcal{N}(r),t) - cn_I(\mathcal{N}(r),t)$, $n_\sigma(\mathcal{N}(r),t)$ represents the total number of particles of species σ in the interaction neighborhood $\mathcal{N}(r)$ at time t, and c is a positive constant.

In order to capture the main characteristics of the activator-inhibitor system discussed in Chapter 2 and gain more control of the system, 5 parameters, p_1, p_2, p_3, p_4, c, are involved. At each cell on the lattice at time t, species A activates itself and species I with the probabilities p_1 and p_3, respectively, when the concentration of species A is greater than the weighted concentration of species I over the interaction neighborhood $\mathcal{N}(r)$. In the meantime, species I inhibits species A and itself with the probabilities p_2 and p_4, respectively, when the concentration of species A is less than the weighted concentration of species I over $\mathcal{N}(r)$. If the concentrations over $\mathcal{N}(r)$ of both species are equal, no destruction or construction of particles will happen. Compared with the original rule, 3 more parameters are introduced and the interaction neighborhood is extended, which gives more freedom of controlling the system. The original rule can be seen as a special case of this modified rule, where $p_1 = p_3$, $p_2 = p_4$, $c = 1$, and $\mathcal{N}(r) = \{r\}$. More different and interesting patterns emerge from this modified rule, as we see later.

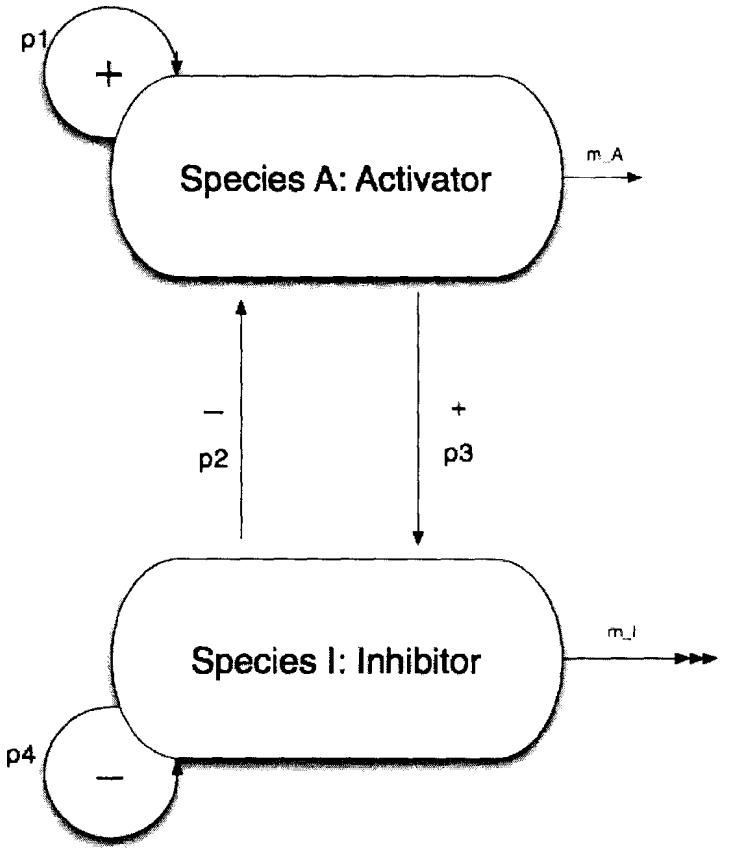

Figure 4.6: An illustration of reaction rule $\mathcal{R}$ for the Activator-Inhibitor System.

4.1.4 An Algorithm of LGCA Model for A-I System

In this section, we present a general algorithm for constructing a LGCA system based on the above description and analysis. The algorithm is as follows:

1. Initialization

2. During the reaction process at time $t = k$, compute the post-reaction states of all channels of species A and I using Reaction rule $\mathcal{R}$ (4.5) and (4.6) and replace the state of each channel of both species by the post-reacction state. Boundary conditions are applied here.

3. During the shuffling process at time $t = k$, compute the post-shuffling states of all channels of species A and I using Shuffling rule $\mathcal{M}$ (4.2) and replace the state of each channel of both species by the post-shuffling state.

4. During the propagation process at time $t = k$, using (4.4), the state of channel $(r + m_\sigma c_i, c_i)$ of species σ is updated by the state of channel (r, c_i) of species σ and then we obtain the new state of each channel at time $t = k + 1$. Boundary conditions are applied here.

5. Let $k = k + 1$ and go to step 2.

In the following sections, we will construct a 1D and 2D LGCA model for the A-I system based on the above algorithm.

4.2 1D LGCA Model for A-I System

In this section, a 1D two-component LGCA model for the activator-inhibitor system is constructed based on the rules and algorithm described in the previous sections. Consider two identical one dimensional lattices $\mathcal{L}_A$ and $\mathcal{L}_I$ for activator A and inhibitor I, respectively, which have the same number of cells $L = |\mathcal{L}_A| = |\mathcal{L}_I|$. Each cell of both species possesses $s = 3$ channels (two velocity channels and one rest channel). Then the state of each cell of species σ is given by $\eta_\sigma(r,t) = (\eta_{\sigma,1}(r,t), \eta_{\sigma,2}(r,t), \eta_{\sigma,3}(r,t))$

at cell r time t which is characterized by $\mathcal{E} = \{0,1\}^3$, where $r = 0, 1, ..., L-1$ and $\sigma = \{A, I\}$. The two velocity channels of species σ are labeled by $(r, c_1)_\sigma$ and $(r, c_2)_\sigma$ with $c_1 = 1$ and $c_2 = -1$,which represents that particles in $(r, c_1)_\sigma$ and $(r, c_2)_\sigma$ will move to the right and left, respectively. The speed of particles of species σ is m_σ and $m_I \geqslant m_A = 1$. The rest channel of species σ is labeled by $(r, c_3)_\sigma)$ with $c_3 = 0$ when the particle residing there will not move. For this model we choose four different interaction neighborhoods $\mathcal{N}_1 = \{r\}$, $\mathcal{N}_2 = \{r, r+1\}$, $\mathcal{N}_3 = \{r-1, r\}$ and $\mathcal{N}_4 = \{r-1, r, r+1\}$ for cell r. Periodic boundary conditions are applied for both species, i.e.,

$$\eta_{\sigma,i}(r,t) = \eta_{\sigma,i}(r+L,t),$$

where $i = 1, 2, 3, r = 0, ..., L-1, t = 0, 1, 2, ..., \sigma = \{A, I\}$.

4.2.1 Numerical Simulations of 1D LGCA

In this section, numerical simulations for different $\mathcal{N}_i(r)$, $i = 1, 2, 3, 4$, and values of the parameters are given. The simulations are computed by Matlab. Our simulations and discussions will be focused on the conditions for generating stationary patterns and the comparison between different $\mathcal{N}_i(r)$ with the same values of the parameters. When there emerges a stationary pattern, the wave number, denoted by k^N, is computed using a discrete Fourier transform numerically in Matlab. We choose the lattice size $L = 100$, the time step $Nt = 300$ and the initial condition is $\eta_{\sigma,i}(r, 0) = (rand > 0.5)$, $\sigma = \{A, I\}, i = 1, 2, 3, r = 0, ..., L-1$, where $rand$ is a random number, chosen from a uniform distribution on the interval $(0, 1)$. This initial condition is randomly generated and saved, so the simulations are using the same initial condition when not stated otherwise.

Firstly, we investigate under what conditions a stationary pattern will emerge. Through the numerical simulations we find that no stationary pattern emerges for all cases if $1 \leqslant m_I \leqslant m_A$. Two examples are given as shown in Figure 4.7 and Figure 4.8. This confirms our analysis since the speed of the inhibitor should be greater than that of the activator. As we increase the value of m_I with fixed value of $m_A = 1$ and

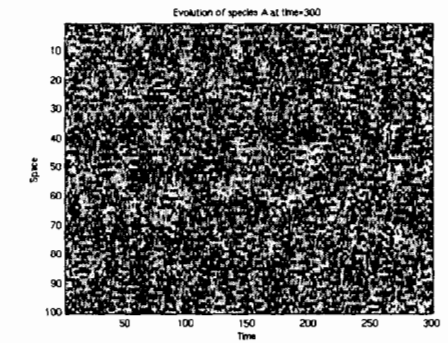

Figure 4.7: Evolution of species A at $t = 300$ with $(m_A, m_I) = (3,3)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

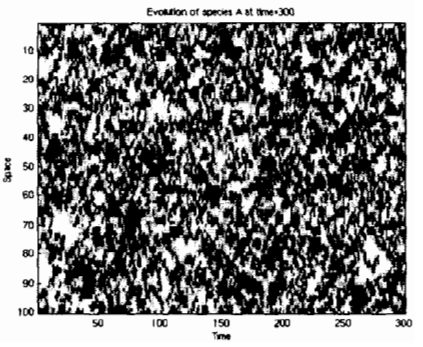

Figure 4.8: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1,1)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

$p = p_i = 1, i = 1, .., 4$, a stationary pattern emerges when $m_I \geqslant 3$ for $\mathcal{N}_1$, $\mathcal{N}_2$ and $\mathcal{N}_3$ and when $m_I \geqslant 2$ for $\mathcal{N}_4$. Four examples are given as shown in Figures 4.9, 4.10, 4.11 and 4.12. From these four examples we can also see that the different patterns emerge for the same values of the parameters but different $\mathcal{N}$.

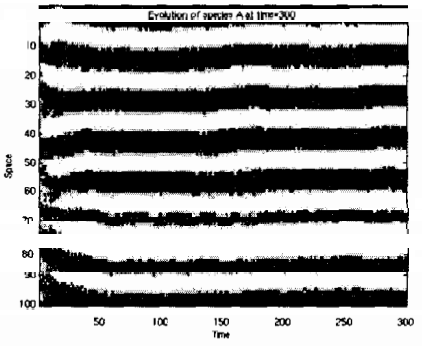

Figure 4.9: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1,8)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 6$.

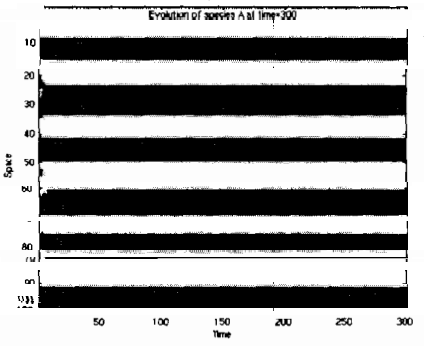

Figure 4.10: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1,8)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 6$.

Secondly, we investigate the relationship between the wave number k^N and the parameters. An approximation was firstly found for $\mathcal{N}_1$ in [1]. We find that this

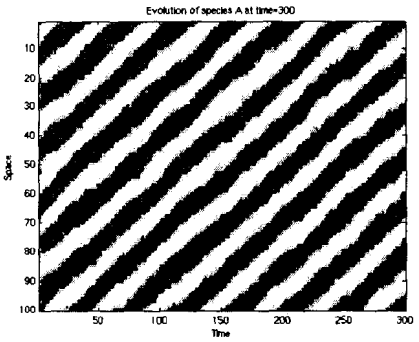

Figure 4.11: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1, 8)$, $\mathcal{N}(r) = \mathcal{N}_2(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 6$.

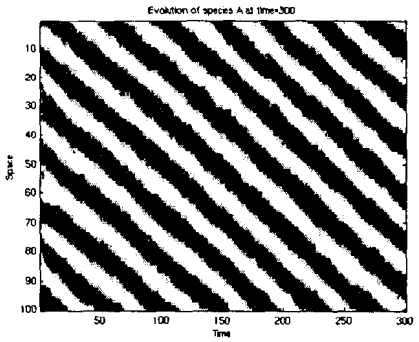

Figure 4.12: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1, 8)$, $\mathcal{N}(r) = \mathcal{N}_3(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 6$.

approximation is also working for $\mathcal{N}_2$ and $\mathcal{N}_3$, i.e., the wave number k^N can be approximated by

$$k^N = [\frac{L}{2m_I}] \qquad \text{for} \qquad m_A = 1, m_I \geqslant 3 \tag{4.7}$$

for $\mathcal{N}_1$, $\mathcal{N}_2$ and $\mathcal{N}_3$, where $[x]$ denotes the integer closest to $x \in \mathbb{R}$. This is also shown in Figures 4.9, 4.11, 4.12, with $k^N = 6 = [\frac{100}{16}]$.

Another thing we find is that for $\mathcal{N}_4$ we always have perfect straight patterns for any values of the parameters but for $\mathcal{N}_1$ we have perfect straight patterns only when k^N is an integer factor of the lattice size L as shown in Figures 4.13 and 4.14.

Thirdly, simulations are performed with $m_I = 2m_A$ for $\mathcal{N}_i, i = 1, 2, 3, 4$. We find that the LGCA shows checkerboard instability for $\mathcal{N}_i, i = 1, 2, 3$. The state of each cell is alternating from 0 to 1, as shown in Figures 4.16 and 4.17. But this instability disappears by extending the interaction neighborhood. When we choose $\mathcal{N}_4$, stationary patterns (perfect straight stripes) emerge instead of checkerboard structure as shown in Figure 4.15.

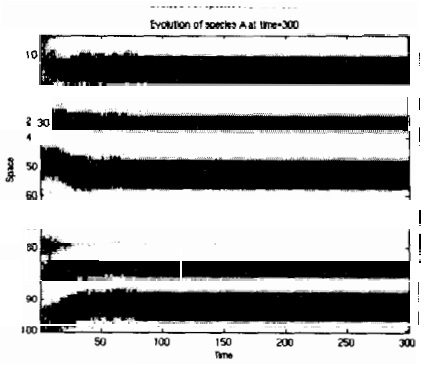

Figure 4.13: Perfect straight stripes of species A at $t = 300$ with $(m_A, m_I) = (1, 10)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 5$.

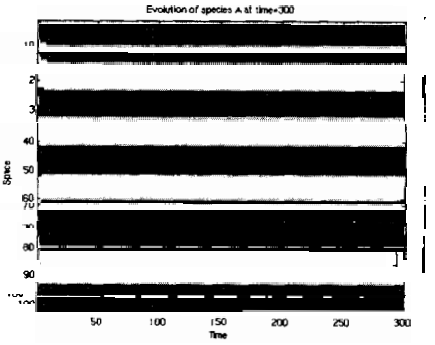

Figure 4.14: Perfect straight stripes of species A at $t = 300$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 6$.

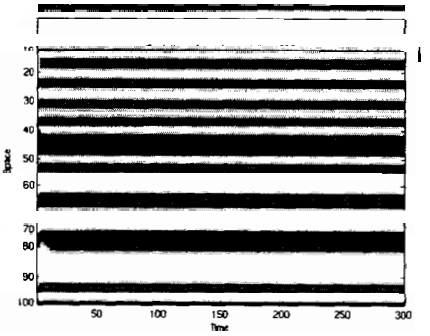

Figure 4.15: Perfect straight stripes of species A at $t = 300$ with $(m_A, m_I) = (1, 2)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 17$.

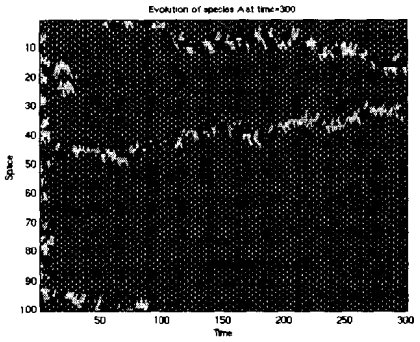

Figure 4.16: Checkerboard structure of species A at $t = 300$ with $(m_A, m_I) = (1, 2)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

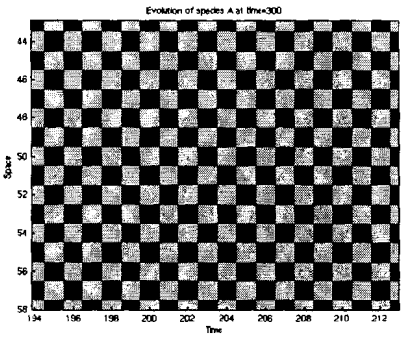

Figure 4.17: Magnified version of the left figure.

Lastly, we investigate how the system performs as $p_i, i = 1, 2, 3, 4$ changes with the other values of the parameters fixed. There are two extreme cases here. The first case is $p_i = 0, i = 1, 2, 3, 4$, which means no reaction occurs. This corresponds to the random walk case, which shows no stationary pattern emerges. The other case is $p_i = 1, i = 1, 2, 3, 4$, which means the reaction process is deterministic. The reaction occurs at each time step at each cell on the lattices. Some examples have been shown above. So what interests us here is when $p_i \neq 0, 1$. In [1] the authors show the existence of a critical $p_c = p_i \approx 0.247, i = 1, 2, 3, 4$ with $(m_A, m_I) = (1, 7), L = 100$. If p is less than p_c, there is less chance to get a stationary pattern by the analysis in [1]. What we find here is that we can still observe very clear stationary patterns when p is very small and close to 0 for $\mathcal{N}_i, i = 2, 3, 4$. We have not found this critical p_c theoretically, however, such critical p must be less than p_c and close to zero based on the numerical simulations. Four simulations are shown in Figures 4.18, 4.20, 4.21 and 4.19, from which we observe that stationary patterns emerge when we choose $\mathcal{N}_i, i = 2, 3, 4$ and no stationary pattern emerges when we choose $\mathcal{N}_1$ with $p = 0.1 < p_c$.

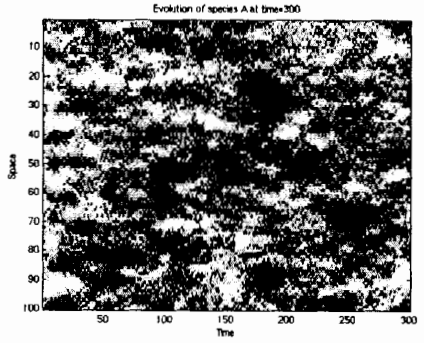

Figure 4.18: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 0.1$.

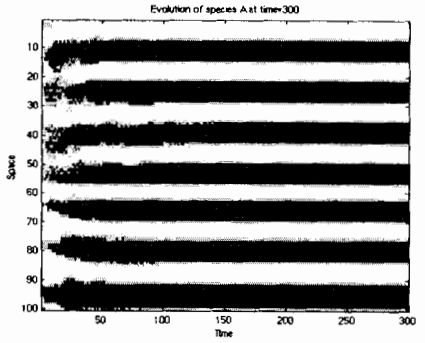

Figure 4.19: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 0.1.k^N = 7$

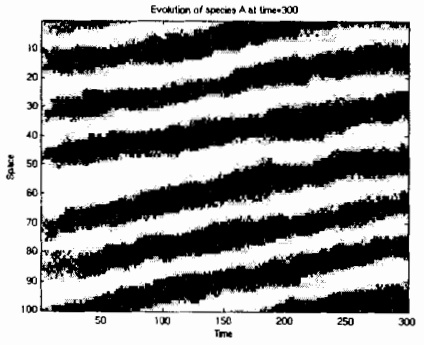

Figure 4.20: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_2(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 0.1.k^N = 6$

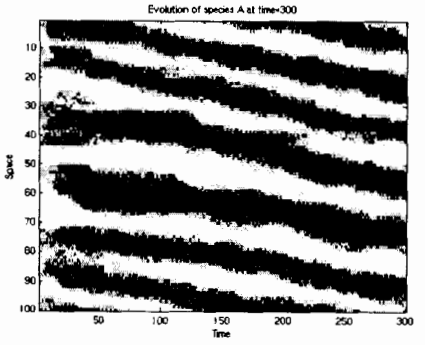

Figure 4.21: Evolution of species A at $t = 300$ with $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_3(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 0.1.k^N = 6$

4.2.2 The Role of Rest Channel in 1D LGCA

In the previous section we observe that the LGCA system with two velocity channels and one rest channel exhibits checkerboard structure when $m_I = 2m_A$ for $\mathcal{N} = \mathcal{N}_1 = \{r\}$. In this section we shall see that the lack of the rest channel can also cause checkerboard instability for almost all values of the parameters, and this instability can be eliminated either by introducing the rest channel or extending the interaction neighborhood.

The presence of a rest channel in the LGCA model sometimes plays a very important role. Here we investigate the LGCA system for the A-I system without the rest channel and compare it with the LGCA model with the rest channel discussed in the previous section. In this model, only two velocity channels are involved. The state of each cell is characterized by $\mathcal{E} = \{0, 1\}^2$. Numerical simulations are performed with the initial condition $\eta_{A,1}(50, 0) = 1$ and $L = 100, Nt = 600$. From Figures 4.22 and 4.23 we observe that the LGCA system shows a checkerboard structure when the rest channel is absent. This checkerboard structure disappears if the rest channel is present as shown in Figure 4.24 with the same initial condition and values of the parameters. The checkerboard structure can also be eliminated by extending the interaction neighborhood $\mathcal{N}_i$ as shown in Figures 4.25, 4.26 and 4.27. We will also investigate this phenomenon for the 2D case in the following sections.

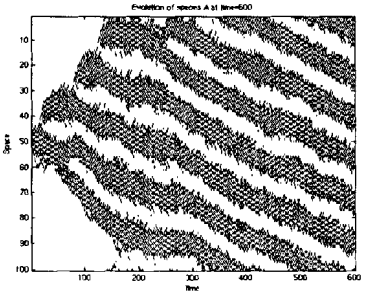

Figure 4.22: Checkerboard structure of species A at $t = 600$ in the LGCA model without rest channel. $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$

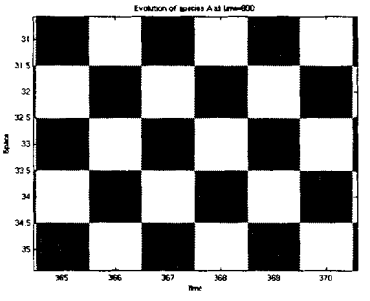

Figure 4.23: Magnified version of the left figure.

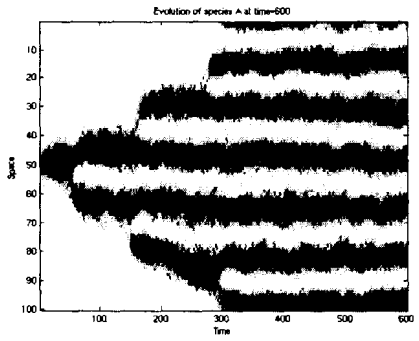

Figure 4.24: Evolution of species A at $t = 600$ in the LGCA model with one rest channel. $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 6$.

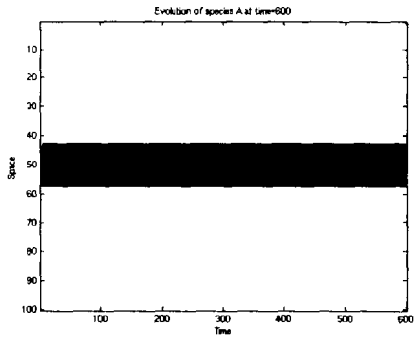

Figure 4.25: Evolution of species A at $t = 600$ in the LGCA model without rest channel. $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 1$.

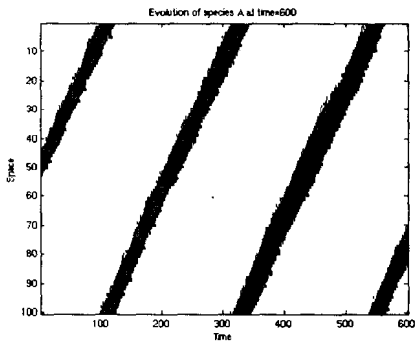

Figure 4.26: Evolution of species A at $t = 600$ in the LGCA model without rest channel. $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_2(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 1$.

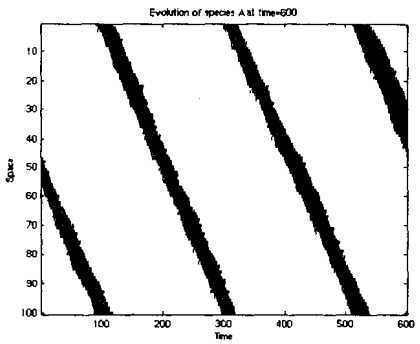

Figure 4.27: Evolution of species A at $t = 600$ in the LGCA model without rest channel. $(m_A, m_I) = (1, 7)$, $\mathcal{N}(r) = \mathcal{N}_3(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$. $k^N = 1$.

4.2.3 The Gray-Scott Model and LGCA Model

In this section, we will see some interesting spatio-temporal dynamics such as self-replication, a standing pulse, and a traveling pulse which are observed in a reaction-diffusion system are also found in the LGCA system. We firstly introduce the Gray-Scott model, and it is then solved numerically using the 2-SBDF method in 1D with

Neumann boundary conditions and different initial conditions and values of the parameters. Numerical solutions are then compared with those obtained from the 1D LGCA model.

The chemical reaction $U + 2V \rightarrow 3V$ and $V \rightarrow P$ in a gel reactor can be described by the following PDEs [11]:

$$\begin{cases} \frac{\partial u}{\partial t} = \rho(1-u) & -u^2v + D_u\nabla^2 u \\ \frac{\partial v}{\partial t} = -(\rho+k)v & +u^2v + D_v\nabla^2 v, \end{cases} \tag{4.8}$$

where u and v are concentrations of the chemical materials U and V, respectively, D_u and D_v are the diffusion coefficients, ρ and k are constants.

This model is first introduced by P. Gray and S.K. Scott and is known as Gray-Scott (GS) model. The GS model shows very rich behaviors for different values of ρ and k, such as self-replication, spikes, traveling waves, spatio-temporal chaos, etc.([11] [12]). Here we omit the details of the discretization of this model and focus on the numerical results and comparisons with the LGCA model. The GS model is discretized using the 2-SBDF method with Neumann boundary conditions. The evolution of species v is plotted in the $t-x$ plane for different values of the parameters and initial conditions.

For studying the self-replicating dynamics, we choose $u(x,0) = 1 - 0.5\sin^{100}(\pi x)$ and $v(x,0) = 0.5\sin^{100}(\pi x)$, a single pulse centered at $x = 0.5$ for each species u and v, $(\rho, k) = (0.04, 0.06)$ and domain size $L = 1$. Figure 4.29 shows that this single pulse keeps splitting and self-relicating, and a stationary pattern with 8 stripes evolves eventually. This type of pattern is also found in the 1D LGCA models with the rest channel and without the rest channel. Here we show an example for the 1D LGCA model with a rest channel. We start with a single seed at channel $(50, c_3)$, i.e., $\eta_3(50,0) = 1$ and the other channels are all 0. We choose $(m_A, m_I) = (1,5)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_i = 1, i = 1,2,3,4$. From Figure 4.28 we observe that this single seed is self-replicating and a stationary pattern with $k^N = 8$ stripes eventually emerges.

For the standing pulse dynamics, we choose $(\rho, k) = (0.05, 0.062)$, and the other conditions remain the same for the GS model. A standing pulse evolves in this case

as shown in Figure 4.31. Similar dynamics are observed in the LGCA model when we change the interaction neighborhood to be $\mathcal{N}_4$ and keep the other conditions unchanged, as shown in Figure 4.30.

For the traveling pulse dynamics, we choose $(\rho, k) = (0.025, 0.0544)$ and $L = 1$ for the GS model. A right traveling pulse and a left traveling pulse, as shown in Figure 4.32 and Figure 4.34, emerge in this model with the initial condition $u(x,0) = 1 - 0.5\sin^{100}(\pi x)$, $v(x,0) = 0.5\sin^{100}(\pi x)$ and $u(x,0) = 1 - 0.5\sin^{100}(\pi(x+0.5))$ and $v(x,0) = 0.5\sin^{100}(\pi(x+0.5))$, respectively. The LGCA model also shows similar dynamics. We choose $\mathcal{N}_2$ and start with a single seed at channel $(0, c_3)$, i.e., $\eta_3(0,0) = 1$, and a right-going pulse emerges as shown in Figure 4.34. Then we choose $\mathcal{N}_3$ and start with a single seed at channel $(100, c_3)$, i.e., $\eta_3(100,0) = 1$, and a left-going pulse emerges as shown in Figure 4.32.

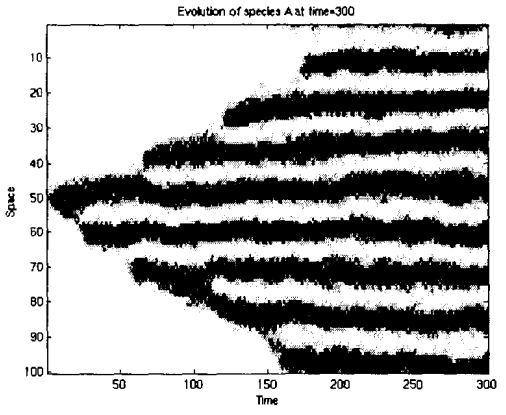

Figure 4.28: Self-replicaton of species A in the LGCA model. $(m_A, m_I) = (1,5)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1, L = 100$. $k^N = 8$.

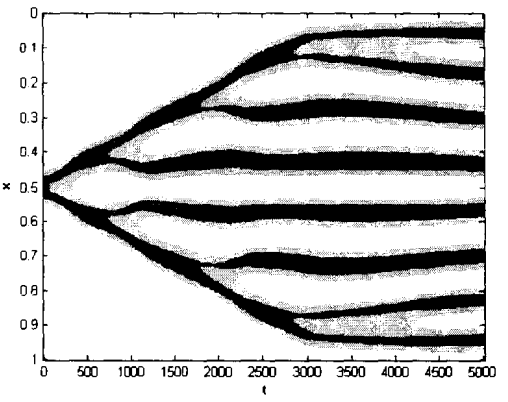

Figure 4.29: Self-replication of v in the GS model. $\rho = 0.04$, $k = 0.06$, $L = 1$.

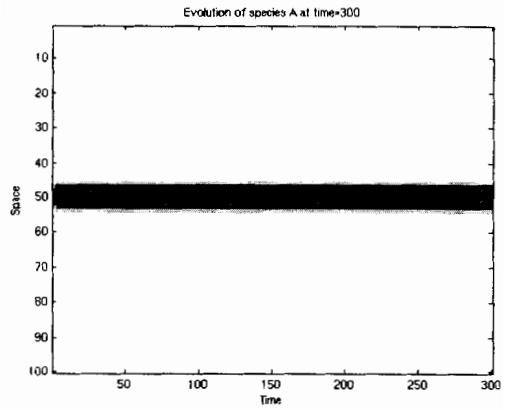

Figure 4.30: Standing pulse of species A in the LGCA model. $(m_A, m_I) = (1, 5)$, $\mathcal{N}(r) = \mathcal{N}_4(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1, L = 100$.

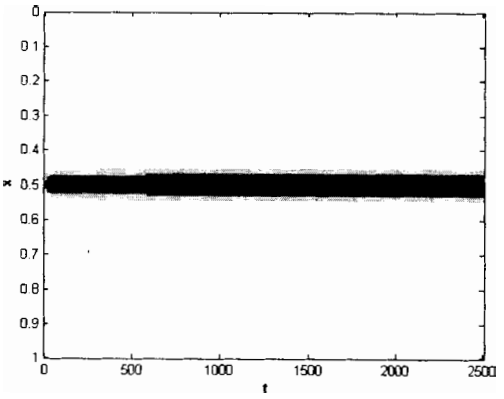

Figure 4.31: Standing pulse of v in the GS model. $\rho = 0.05$, $k = 0.062$, $L = 1$.

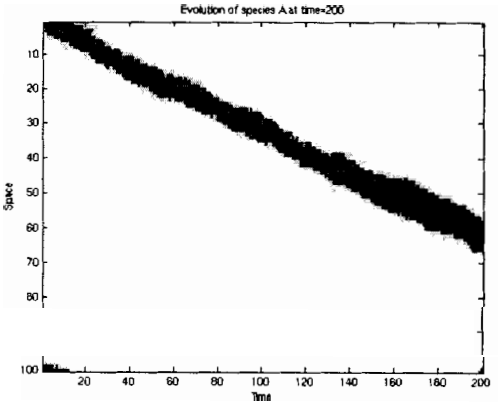

Figure 4.32: Right traveling pulse of species A in the LGCA. $(m_A, m_I) = (1, 5)$, $\mathcal{N}(r) = \mathcal{N}_2(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1, L = 100$.

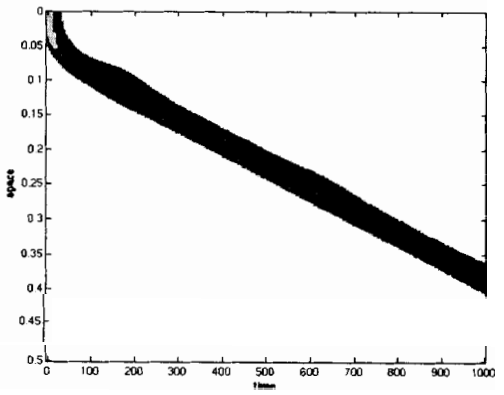

Figure 4.33: Right traveling pulse of v in the GS model. $\rho = 0.025$, $k = 0.0544$, $L = 0.5$.

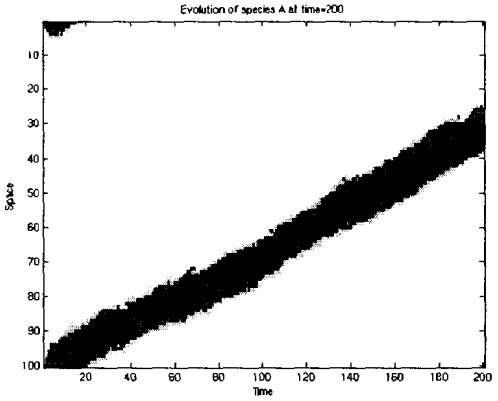

Figure 4.34: Left traveling pulse of species A in the LGCA. $(m_A, m_I) = (1,5), \mathcal{N}(r) = \mathcal{N}_3(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1, L = 100$.

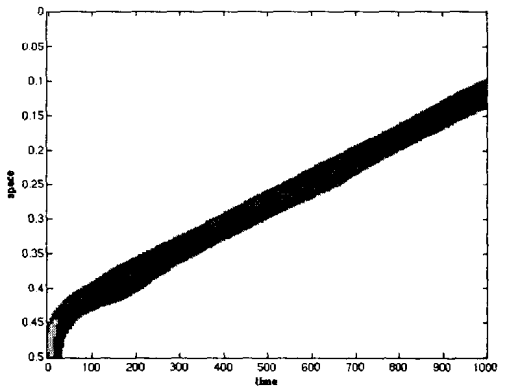

Figure 4.35: Left traveling pulse of v in the GS model. $\rho = 0.025$, $k = 0.0544$, $L = 0.5$.

4.3 2D LGCA Model for A-I System

In this section we extend our analysis and simulations to two dimensions. A 2D two-component LGCA model for an activator-inhibitor system is constructed based on the updating rules and algorithm described in the previous sections. Consider two identical two dimensional lattices $\mathcal{L}_A$ and $\mathcal{L}_I$ for activator A and inhibitor I, respectively, which have the same number of cells $L^2 = |\mathcal{L}_A| = |\mathcal{L}_I|$. Each cell of both species possesses $s = 5$ channels (four velocity channels and one rest channel). Then the state of a cell of species σ is given by a Boolean vector $\eta_\sigma(r,t) = (\eta_{\sigma,1}(r,t), \eta_{\sigma,2}(r,t), \eta_{\sigma,3}(r,t), \eta_{\sigma,4}(r,t), \eta_{\sigma,5}(r,t))$ at cell r time t which is characterized by $\mathcal{E} = \{0,1\}^5$, where $r = (x,y)$ with $x = 0,...,L-1, y = 0,...,L-1$. The four velocity channels are labeled by $(r,c_1)_\sigma$, $(r,c_2)_\sigma$, $(r,c_3)_\sigma$ and $(r,c_4)_\sigma$ with $c_1 = 1, c_2 = 1, c_3 = -1, c_4 = -1$, which represents that a particle in $(r,c_1)_\sigma$, $(r,c_2)_\sigma$, $(r,c_3)_\sigma$,$(r,c_4)_\sigma$ will move to the right, up, left, and down, respectively. The speed of particles of species σ is m_σ with $m_I \geqslant m_A = 1$. The rest channel of species σ is labeled by $(r,c_5)_\sigma$ with $c_5 = 0$ when the particle residing there will not move. For this model we choose three different interaction neighborhoods, $\mathcal{N}_1 = \{r\}$, $\mathcal{N}_2 = \{r, r+(0,1), r+(0,-1), r+(-1,0), r+(1,0)\}$ and $\mathcal{N}_3 = \mathcal{N}_2/\{r\} = \{r+$

$(0,1), r+(0,-1), r+(-1,0), r+(1,0)\}$. Periodic boundary conditions are applied for both species, i.e.,

$$\begin{cases} \eta_{\sigma,i}(x,y,t) = \eta_{\sigma,i}(x+L,y,t) \\ \eta_{\sigma,i}(x,y,t) = \eta_{\sigma,i}(x,y+L,t),. \end{cases} \tag{4.9}$$

where $i = 1,2,3,4,5$, $x,y = 0,...,L-1, t = 0,1,2,....., \sigma = \{A,I\}$.

4.3.1 Numerical Simulations of 2D LGCA

In this section, numerical simulations for different $\mathcal{N}_i(r)$, $i = 1,2,3$, and values of the parameters are performed. The simulations are computed by Matlab. Our simulations and discussions will be focused on the comparison between the 1D LGCA model and 2D LGCA model in pattern formation, the comparison between different $\mathcal{N}_i(r)$ and some interesting patterns generated with some special values of parameter c introduced in the reaction rules 4.5 and 4.6. We choose lattice size $L = L_x = L_y = 100$, time step $Nt = 100$, and the initial condition is a single seed at channel $((50,50),5)_A$, i.e.,$\eta_{A,5}(50,50,0) = 1$. If not stated otherwise, the simulations will start with the same initial condition.

Firstly, we investigate the pattern formation in 2D. We find that no stationary pattern emerges if $m_I = m_A$ or $m_I = 2m_A$ for all $\mathcal{N}_i$, $i = 1,2,3$. This is a little different from the results in 1D, where if $m_I = 2m_A$, the LGCA is still capable of generating a stationary pattern when $\mathcal{N}$ is extended to $\{r-1, r, r+1\}$. As we increase the value of m_I, a stationary pattern emerges as shown in Figures 4.36, 4.37, 4.38 and 4.39.

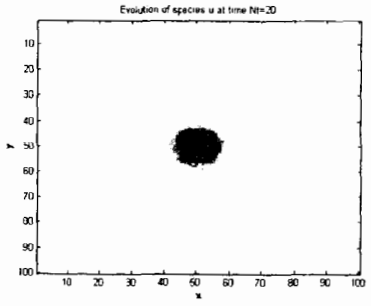

Figure 4.36: Evolution of activator A at $t = 20$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_1$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

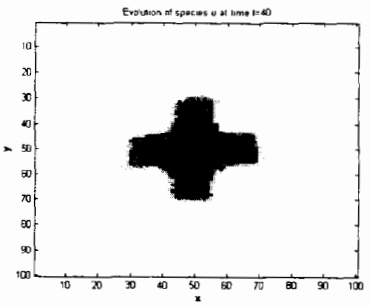

Figure 4.37: Evolution of activator A at $t = 40$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_1$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

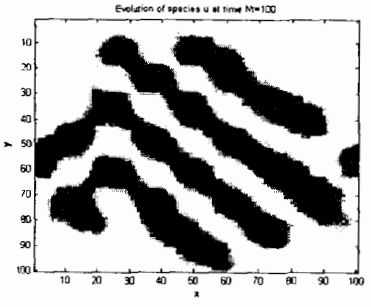

Figure 4.38: Evolution of activator A at $t = 100$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_1$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

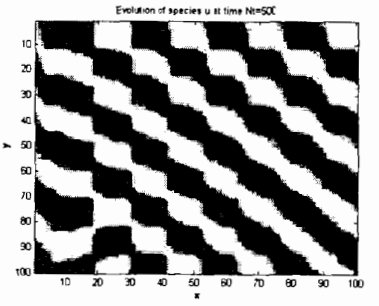

Figure 4.39: Evolution of activator A at $t = 500$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_1$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

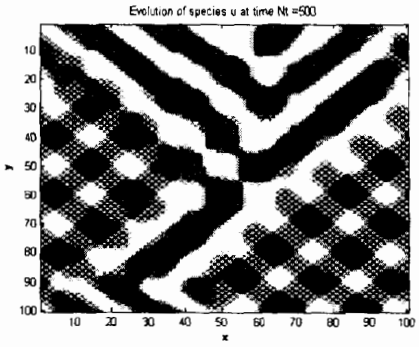

Figure 4.40: Evolution of activator A at $t = 500$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_3$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

Secondly, we investigate the system with different $\mathcal{N}_i$. Different patterns emerge for different $\mathcal{N}_i$ as shown in Figures 4.41, 4.42 and 4.40. Comparing Figure 4.39 and Figure 4.41, we also observe that different patterns emerge even when the initial configurations are the same. If we rotate Figure 4.39 by 90 degrees counter-clock wise we essentially obtain Figure 4.41.

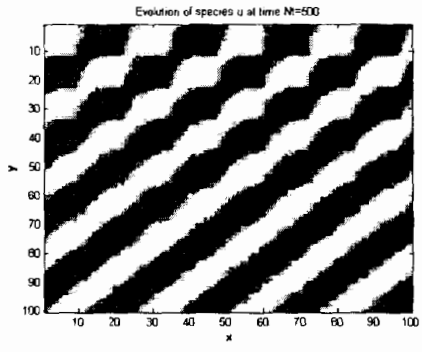

Figure 4.41: Evolution of activator A at $t = 500$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_1$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

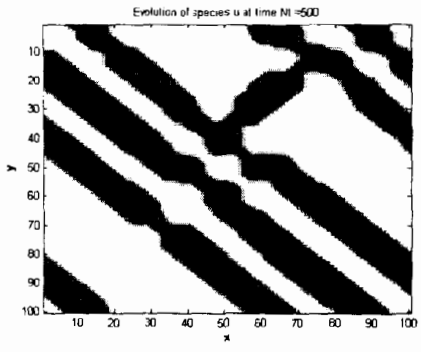

Figure 4.42: Evolution of activator A at $t = 500$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_2$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

Lastly, we find that symmetric patterns emerge as we change the value of c in the 2D LGCA system. An example of the evolution of a symmetric stationary pattern is shown in Figures 4.43, 4.44, 4.45 and 4.46.

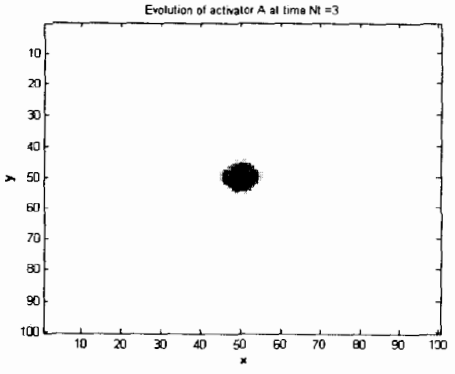

Figure 4.43: Evolution of activator A at $t = 3$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_2$ and $c = 1.1, p_1 = p_2 = p_3 = p_4 = 1$.

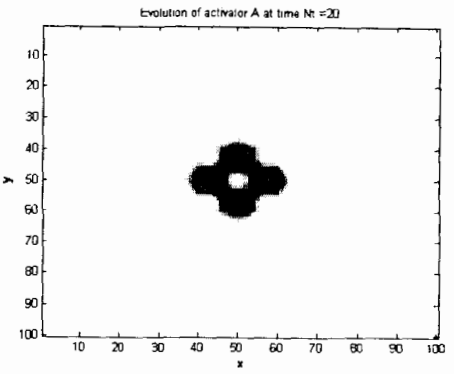

Figure 4.44: Evolution of activator A at $t = 20$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_2$ and $c = 1.1, p_1 = p_2 = p_3 = p_4 = 1$.

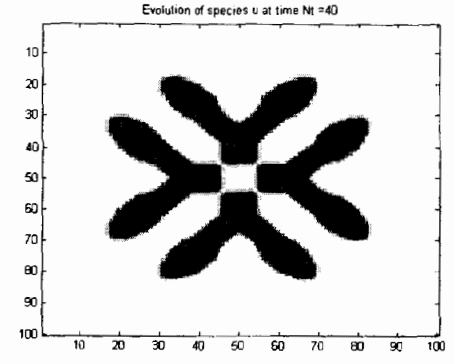

Figure 4.45: Evolution of activator A at $t = 40$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_2$ and $c = 1.1, p_1 = p_2 = p_3 = p_4 = 1$.

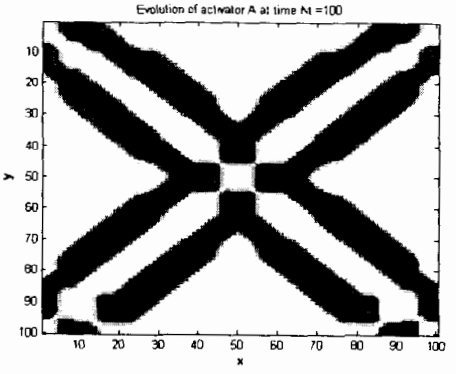

Figure 4.46: Evolution of activator A at $t = 100$ with $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_2$ and $c = 1.1, p_1 = p_2 = p_3 = p_4 = 1$.

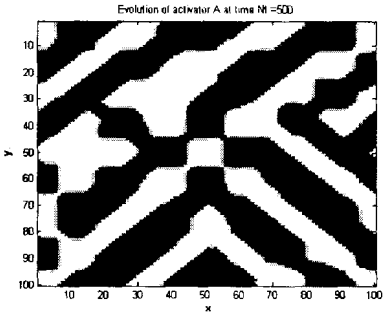

Figure 4.47: Evolution of activator A at $t = 500$ in the LGCA model without rest channel. $(m_A, m_I) = (1, 9)$, $\mathcal{N} = \mathcal{N}_2$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$.

4.3.2 The Role of Rest Channel in 2D LGCA

Similarly to the 1D case, the absence of the rest channel in the 2D LGCA system may cause a checkerboard structure, and such structure can be eliminated by extending the interaction neighborhood. Figures 4.48 and 4.49 show that we still get the stripe-like stationary pattern for the LGCA model without a rest channel, but the checkerboard structure is observed inside the stripes. Figure 4.47 shows this structure disappear if $\mathcal{N} = \mathcal{N}_2$. The same values of the parameters are chosen for these three simulations.

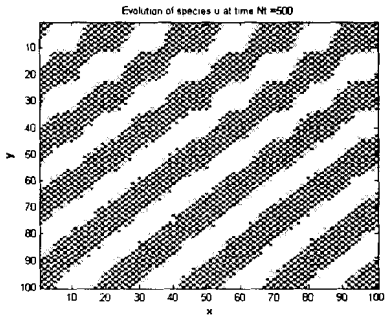

Figure 4.48: Checkerboard structure of activator A at $t = 500$ in the LGCA model without the rest channel. $(m_A, m_I) = (1, 9)$, $\mathcal{N}(r) = \mathcal{N}_1(r)$ and $c = 1, p_1 = p_2 = p_3 = p_4 = 1$

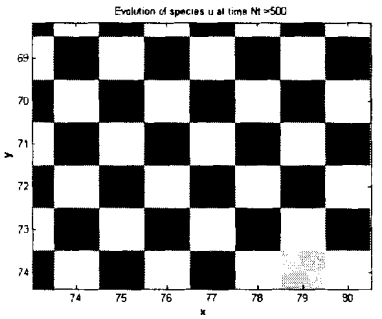

Figure 4.49: Magnified version of the left figure.

4.4 Commutativity of Updating Operators

The above simulations and discussions are based on the fact that the three updating operators are in the order of $\mathcal{P} \circ \mathcal{M} \circ \mathcal{R}$, where $\mathcal{R}$ is the first. In this section, we investigate the commutativity of these three operators through numerical simulations.

Firstly, we investigate the commutativity of $\mathcal{P}$ and $\mathcal{M}$. In section 4.1.1 we discuss the random walk of a particle by the $\mathcal{PM}$ dynamics. Since the dynamics of the LGCA system arises from the repetitive applications of these two operators, if we let $\mathcal{D}_1 = \mathcal{PM}$ and $\mathcal{D}_2 = \mathcal{MP}$, after n iterations we have

$$\mathcal{D}_2^n = \mathcal{M}\mathcal{D}_1^{n-1}\mathcal{P}. \tag{4.10}$$

Since $\mathcal{M}$ does not change the spatial position of a particle (a particle residing on cell r is still in cell r after the shuffling process) and a particle residing on the rest channel at $t = 0$ will not propagate at the first time step, then (4.10) can be reduced to

$$\mathcal{D}_2^n = \mathcal{D}_1^{n-1}. \tag{4.11}$$

So a particle will also perform a random walk by the $\mathcal{MP}$ dynamics. The mean distance and the probability density function for $\mathcal{MP}$ dynamics are computed as shown in Figures 4.50 and 4.51.

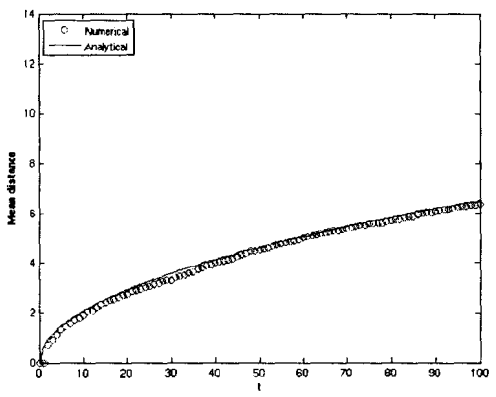

Figure 4.50: Mean distance in the $\mathcal{MP}$ dynamics.

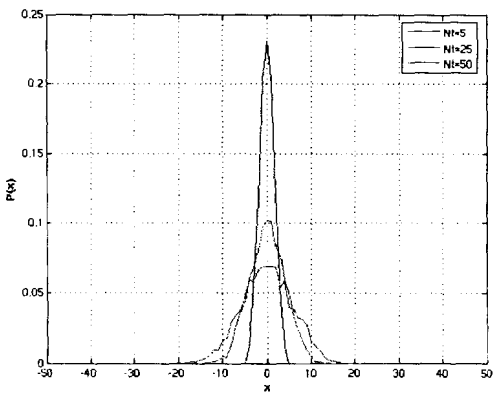

Figure 4.51: Gaussian Structure in the $\mathcal{MP}$ dynamics.

Secondly, we investigate the commutativity of $\mathcal{D}_1$ and $\mathcal{R}$ through numerical simulations in 1D. A 1D LGCA model with $\mathcal{D}_1\mathcal{R}$ dynamics and $\mathcal{R}\mathcal{D}_1$ dynamics is constructed. The numerical wave number is also calculated for each case. From Figure 4.53 and Figure 4.52 we observe that these two LGCA systems give similar structure and the same wave number k^N. Lastly, the LGCA system also shows qualitatively

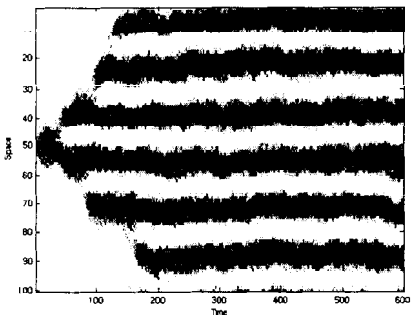

Figure 4.52: Stationary Patterns in the LGCA system with $\mathcal{R}\mathcal{D}_1$ dynamics. $k^N = 6$.

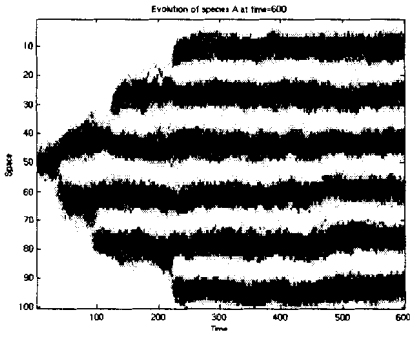

Figure 4.53: Stationary Patterns in the LGCA system with $\mathcal{D}_1\mathcal{R}$ dynamics. $k^N = 6$.

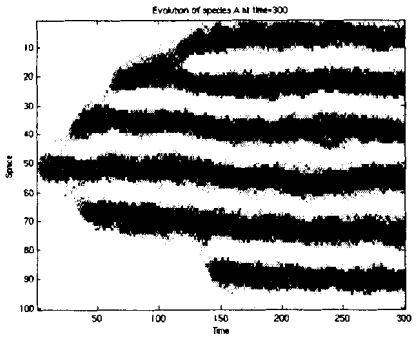

Figure 4.54: Stationary Patterns in the LGCA system with $\mathcal{M}\mathcal{R}\mathcal{P}$ dynamics. $k^N = 6$.

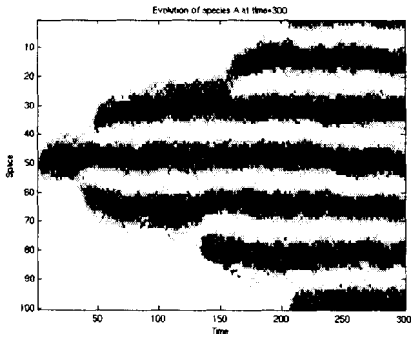

Figure 4.55: Stationary Patterns in the LGCA system with $\mathcal{P}\mathcal{R}\mathcal{M}$ dynamics. $k^N = 6$.

similar results as $\mathcal{R}$ and $\mathcal{M}$ or $\mathcal{R}$ and $\mathcal{P}$ exchange, i.e., the LGCA system with $\mathcal{P}\mathcal{R}\mathcal{M}$ dynamics or $\mathcal{M}\mathcal{R}\mathcal{P}$ dynamics is still capable of generating Turing-type patterns as shown in Figures 4.54 and 4.55.

In this chapter, we have constructed a LGCA model in 1D and 2D for an activator-inhibitor system based on a modified reaction rule $\mathcal{R}$. It should be noticed that only 3 or 4 states are used for each cell in the 1D LGCA (corresponding to the LGCA model without a rest channel and with a rest channel) and 5 or 6 states for each cell in the 2D LGCA (corresponding to the LGCA model without a rest channel and with a rest channel). Despite the simplicity of construction, this model shows very rich dynamics and patterns such as Turing patterns, traveling wave, standing wave, self-replication, symmetric stationary patterns and checkerboard patterns in 1D and 2D based on the reaction rule we defined. From the numerical simulations and analysis, we have found that the dynamics of this LGCA model are characterized not only by the parameters in the local reaction and diffusion rule, but the interaction neighborhood and the rest channel. Without a rest channel, a LGCA model may easily show checkerboard instabilities. We also have found that such instability can be removed by introducing a new neighborhood. We investigated four different interaction neighborhoods for the 1D LGCA and three for the 2D LGCA. Different dynamics arise for these interaction neighborhoods, such as a traveling wave, a standing wave, or self-replication in 1D, which shows qualitatively similar results to the Gray-Scott model. The commutativity of the three operators $\mathcal{P}$, $\mathcal{M}$ and $\mathcal{R}$ is also studied through the numerical simulations. It turns out that the order of these three operators does not affect the global and long time behavior of the LGCA system. All the numerical simulations are performed in Matlab, and the 1D and 2D LGCA (with one rest channel) codes can be found in the Appendix.

Chapter 5

Conclusions

In this thesis, we present a MACA model and a LGCA model for investigating Turing pattern formation in an activator-substrate system and an activator-inhibitor system. In the framework of CA, the Turing patterns can be seen as the emergent behaviors of the system due only to local interactions between cells.

One of the keys of the CA modeling for reaction-diffusion systems is the construction of a reaction rule and a diffusion rule. For the MACA model, the diffusion rule comes from a special discretization of the diffusion equation and the reaction rule comes from a finite-difference-like discretization of the reaction-diffusion PDEs [8]. The diffusive step is implemented by a moving-average procedure based on the discretization. A probabilistic truncation rule is introduced to make the state of each cell represented only by an integer in $[0, 10]$, i.e., there are 11 states in the CA model represented by integers from 0 to 10. We then apply this method to the 2D Brusselator to investigate the Hopf bifurcation and Turing bifurcation. Numerical simulations show that this 11-state CA model can capture the Hopf bifurcation and Turing bifurcation of the Brusselator in a qualitatively correct way compared with the results obtained by solving the reaction-diffusion PDEs using an IMEX scheme, Second Order Backward Differentiation Formula [9].

For the LGCA model, with the help of the introduction of channels for each cell and the propagation of particles residing in corresponding channels, the diffusion rule can be constructed by combining the Propagation rule $\mathcal{P}$ and Shuffling rule $\mathcal{M}$ [1][7][10].

We numerically show that each particle performs a random walk and the movement of each particle displays a Gaussian structure in the non-reactive LGCA system. We then define a Reaction rule $\mathcal{R}$ for an activator-inhibitor system which is based on the rule defined in [1]. This modified reaction rule $\mathcal{R}$ incorporates the original rule and the LGCA system is able to show more complex and interesting patterns with rule $\mathcal{R}$ in 1D and 2D. Complete dynamics of the LGCA system is realized by the repetitive applications of these three operators applied to each cell simultaneously. Based on these rules, we construct a LGCA model with different lattice $\mathcal{L}$, state set $\mathcal{E}$ and interaction neighborhood $\mathcal{N}$. It turns out that the LGCA model (3-state or 4-state) in 1D and the LGCA model (5-state or 6-state) in 2D are able to generate Turing-type patterns and other kinds of instability such as checkerboard structure. The conditions for Turing pattern formation in the LGCA model are discussed and given. Last, we focus on how this LGCA system is characterized by the lattice $\mathcal{L}$, state set $\mathcal{E}$ and interaction neighborhood $\mathcal{N}$. We investigate the Turing patterns in the 1D LGCA and 2D LGCA systems, the role of the rest channel in the LGCA system, and the influences of interaction neighborhood. We find that the absence of the rest channel may cause a checkerboard instability in 1D and 2D LGCA systems and such instability can be eliminated by extending or changing the interaction neighborhood. Comparisons between the dynamics of a 1D 4-state LGCA model and the 1D Gray-Scott model are given and discussed. Structures found in the 1D Gray-Scott model, such as traveling waves, standing waves and self-reproductions, are also found in this 1D LGCA model. Although our simulations and discussions are based on the order of $\mathcal{PMR}$, we argue that in fact these three operators are commutative through the numerical simulations.

Our simulations, for the Brusselator problem and the Gray-Scott problem have been largely phenomenological, and the rich solution behaviors have often been found only after extensive experiments. Nevertheless, we believe that the ability of CA models to qualitatively capture the solution behaviors of reaction-diffusion PDEs shows sufficient promise to warrant a more systematic, analytical investigation.

Chapter 6

Appendices

In this Chapter, some matlab codes for the MACA and the LGCA are attached.

6.1 MACA Code

In this section, a matlab code of the MACA for the 2D Brusselator is shown below.

```
%MACA driver for 2D Brussellator
%Periodic BCs
clf
clear all
close all
clc
global aa
global bb
%Initialization
dt =.01;          % Time step
R_u = 1;          %(3X3) neighbourhood for u
R_v = 2;          %(5X5) neighbourhood for v
n = 250  ;        % Number of cells
a_u = 1/((2*R_u+1)^2);
a_v = 1/((2*R_v+1)^2);
Mu =10; Mv = 10;      % Discretization level,usually in [0,100]
z=zeros(n,n); zz=ones(n,n);
cells_u=z; Hsum_u=z; Vsum_u=z; temp_u = z;
cells_v=z; Hsum_v=z; Vsum_v=z; temp_v = z;
% Initial Conditions
aa = 3;bb=12;
cells_u = aa*zz+.1*rand(n,n);
cells_v = (bb/aa)*zz+.1*rand(n,n);
```

```
imagesc(cells_u)
pause(1)
%%%%%%%%%%%%%%%%%%%%%%%%
for iter=1: 10000
    time = iter*dt
%Compute horizontal sum
  %%%%%%%%%%%%%%%%%%%%
  for i = 1:n  % i represent the y-axis
    Hsum_u(i,2) = cells_u(i,1) + cells_u(i,2)+ cells_u(i,3) ;
    Hsum_v(i,3) = cells_v(i,1) + cells_v(i,2)+ cells_v(i,3) + cells_v(i,4)+
cells_v(i,5) ;

    for j = 3:n-1
        Hsum_u(i,j) = Hsum_u(i,j-1 ) + cells_u(i,j+1) - cells_u(i,j-2);
    end
    for j = 4:n-2
        Hsum_v(i,j) = Hsum_v(i,j-1 ) + cells_v(i,j+2) - cells_v(i,j-3);
    end

      Hsum_u(i,1) = cells_u(i,1) + cells_u(i,2) + cells_u(i,n);
      Hsum_u(i,n) = cells_u(i,n) + cells_u(i,n-1) + cells_u(i,1);

      Hsum_v(i,1) = cells_v(i,1) + cells_v(i,2) + cells_v(i,3) +
cells_v(i,n-1) + cells_v(i,n);
      Hsum_v(i,2) = cells_v(i,2) + cells_v(i,3) + cells_v(i,4) +
cells_v(i,1)   + cells_v(i,n);
      Hsum_v(i,n-1) = cells_v(i,n-1) + cells_v(i,n-2) + cells_v(i,n-3) +
cells_v(i,n)   + cells_v(i,1);
      Hsum_v(i,n) = cells_v(i,n) + cells_v(i,n-1) + cells_v(i,n-2) +
cells_v(i,1)   + cells_v(i,2);
  end
  %%%%%%%%%%%%%%%%%%%%%%%%
  %Compute vertical sum

  for j = 1:n

      Vsum_u(2,j) = Hsum_u(1,j)+ Hsum_u(2,j)+ Hsum_u(3,j);
      Vsum_v(3,j) = Hsum_v(1,j)+ Hsum_v(2,j)+ Hsum_v(3,j) + Hsum_v(4,j)+
Hsum_v(5,j);
      for i = 3:n-1
          Vsum_u(i,j)=Vsum_u(i-1,j) + Hsum_u(i+1,j) - Hsum_u(i-2,j);  % u
>From row 3 to row n-1
      end

      for i = 4:n-2
          Vsum_v(i,j)=Vsum_v(i-1,j) + Hsum_v(i+2,j) - Hsum_v(i-3,j);   % v
>From row 4 to row n-2
```

```
      end

      Vsum_u(1,j) = Hsum_u(1,j) + Hsum_u(2,j) + Hsum_u(n,j);   % u row 1
      Vsum_u(n,j) = Hsum_u(n,j) + Hsum_u(n-1,j) + Hsum_u(1,j); % u row n

      Vsum_v(1,j) = Hsum_v(1,j) + Hsum_v(2,j) + Hsum_v(3,j) +  Hsum_v(n,j) +
Hsum_v(n-1,j);
      Vsum_v(2,j) = Hsum_v(1,j) + Hsum_v(2,j) + Hsum_v(3,j) +  Hsum_v(4,j) +
Hsum_v(n,j);
      Vsum_v(n-1,j) = Hsum_v(n,j) + Hsum_v(n-1,j) + Hsum_v(n-2,j) +
Hsum_v(n-3,j) + Hsum_v(1,j);
      Vsum_v(n,j) = Hsum_v(n,j) + Hsum_v(n-1,j) + Hsum_v(n-2,j) +
Hsum_v(1,j) + Hsum_v(2,j);
end

  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % Construct look-up tables
   temp_u(1:n,1:n) =a_u*Vsum_u(1:n,1:n) +
dt*fuv(cells_u(1:n,1:n),cells_v(1:n,1:n));
   temp_v(1:n,1:n) =a_v*Vsum_v(1:n,1:n) +
dt*guv(cells_u(1:n,1:n),cells_v(1:n,1:n));

  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    X_u = floor(temp_u);
    P_u = temp_u - X_u ;
    X_v = floor(temp_v);
    P_v = temp_v - X_v ;

    for iii= 1:n
        for jjj = 1:n
            Random = rand;
            if          Random <= P_u(iii,jjj)
                    cells_u(iii,jjj) = X_u(iii,jjj) +1;
            else
                    cells_u(iii,jjj) = X_u(iii,jjj);
            end
        end
    end
    for iii = 1:n
        for jjj = 1:n
            Random = rand;
             if          Random <= P_v(iii,jjj)
                    cells_v(iii,jjj) = X_v(iii,jjj) +1;
            else
                    cells_v(iii,jjj) = X_v(iii,jjj);
            end
```

```
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %plot the results
    if mod(iter,5*10)==0
image(cells_u,'cdatamapping','scaled');
    drawnow
    colorbar
   colormap([0 0 0 ; .5 .5 .5; .8 .8 .8;.9 .9 .9 ])
    title( [ ' u at time t=', num2str(time,'%9.4f')] );
   xlabel('\bf x')
   ylabel('\bf y')
 end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [fuv] = fuv(u,v)
global aa
global bb
fuv = u.^2.*v - ( bb + 1)*u + aa;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [guv] = guv(u,v)
global aa
global bb
guv = bb*u - u.^2.*v;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

6.2 1D LGCA Codes

In this section, the matlab codes for the 1D LGCA (with two velocity channels and one rest channel) are attached.

6.2.1 Propagation Rule

```
% Propagation rule (2.4)
% Input: 1. The states of all cells before propagation
%        2. Propagation speed m
% Output: The states of all cells after propagation
%##############################
function [Deta]=PROP(Meta,m)
```

```
 N = length(Meta);
for r = 1:N-m
    Deta(r+m,1) = Meta(r,1);
end
for r = 1:m
    Deta(r,1) = Meta(N-m+r,1);
end
%############################
for r = m+1:N
Deta(r-m,2) = Meta(r,2);
end
for r = 1:m
    Deta(N-m+r,2) = Meta(r,2);
end
%############################
for r = 1:N
Deta(r,  3) = Meta(r,3);
end
```

6.2.2 Shuffling Rule

```
%Shuffling Rule (4.2)
% Input: The states of all cells before shuffling.
% Output: The states of all cells after shuffling.
function [Meta] = SHUFF(eta)
N = length(eta);
for r = 1:N
  ksi=zeros(1,6);
  i = ceil(rand*6);
  ksi(i) = 1;
  ksi_1 = ksi(1);ksi_2= ksi(2);ksi_3= ksi(3);ksi_4 = ksi(4);ksi_5= ksi(5);ksi_6 = ksi(6);

  Meta(r,1) = ksi_1*eta(r,1)+ksi_2*eta(r,1)+ksi_3*eta(r,2)+ksi_4*eta(r,2)+
              ksi_5*eta(r,3)+ksi_6*eta(r,3);
  Meta(r,2) = ksi_1*eta(r,2)+ksi_2*eta(r,3)+ksi_3*eta(r,1)+ksi_4*eta(r,3)+
              ksi_5*eta(r,2)+ksi_6*eta(r,1);
  Meta(r,3) = ksi_1*eta(r,3)+ksi_2*eta(r,2)+ksi_3*eta(r,3)+ksi_4*eta(r,1)+
              ksi_5*eta(r,1)+ksi_6*eta(r,2);
end
```

6.2.3 Reaction Rule 1

```
%1D Reaction rule (4.5) and (4.6) interaction neighborhood N1={r}.
```

```
%Input: 1. The states of all cells of species A and I before reactions.
%       2. Probability p (vector) and a constant c which control the reactions.
%Output: The states of all cells of species A and species I after reactions.

function [Reta1, Reta2] = REA1(eta1,eta2,c,p)
N=length(eta1);
nA =zeros(N,1);
nI =zeros(N,1);
Reta1=zeros(N,3);
Reta2=zeros(N,3);
Vone = [1 1 1];
Oone = [0 0 0];
nA(:) = eta1(:,1)+eta1(:,2)+eta1(:,3);
nI(:) = eta2(:,1)+eta2(:,2)+eta2(:,3);
Diff = nA - c*nI;

  for r = 1:N

      if Diff(r)>0 & (rand>(1-p(1)))
           Reta1(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(2)))
           Reta1(r,:) = Oone(:);
      else
          Reta1(r,:) = eta1(r,:);
      end

        if Diff(r)>0 & (rand>(1-p(3)))
           Reta2(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(4)))
           Reta2(r,:) = Oone(:);
      else
          Reta2(r,:) = eta2(r,:);
        end
   end
```

6.2.4 Reaction Rule 2

```
%1D Reaction rule (4.5) and (4.6) for interaction neighborhood N2={r,r+1}.
%Input: 1. The states of all cells of species A and I before reactions.
%       2. Probability p (vector) and a constant c which control the reactions.
%Output: The states of all cells of species A and species I after reactions.

function [Reta1, Reta2] = REA2(eta1,eta2,c,p)
N=length(eta1);
nA =zeros(N,1);
```

```
nI =zeros(N,1);
Reta1=zeros(N,3);
Reta2=zeros(N,3);
Vone = [1 1 1];
Done = [0 0 0];
for r = 1:N-1
  nA(r) =eta1(r,1)+eta1(r,2)+eta1(r,3)+eta1(r+1,1)+eta1(r+1,2)+eta1(r+1,3);
  nI(r) =eta2(r,1)+eta2(r,2)+eta2(r,3)+eta2(r+1,1)+eta2(r+1,2)+eta2(r+1,3);
end
  nA(N) = eta1(N,1)+eta1(N,2)+eta1(N,3)+eta1(1,1)+eta1(1,2)+eta1(1,3);
  nI(N) = eta2(N,1)+eta2(N,2)+eta2(N,3)+eta2(1,1)+eta2(1,2)+eta2(1,3);
  Diff = nA - c*nI;

  for r = 1:N

      if Diff(r)>0 & (rand>(1-p(1)))
           Reta1(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(2)))
           Reta1(r,:) = Done(:);
      else
          Reta1(r,:) = eta1(r,:);
      end

        if Diff(r)>0 & (rand>(1-p(3)))
           Reta2(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(4)))
           Reta2(r,:) = Done(:);
      else
          Reta2(r,:) = eta2(r,:);
        end
   end
```

6.2.5 Reaction Rule 3

```
%1D Reaction rule (4.5) and (4.6) for interaction neighborhood N3={r-1,r}.
%Input: 1. The states of all cells of species A and I before reactions.
%       2. Probability p (vector) and a constant c which control the reactions.
%Output: The states of all cells of species A and species I after reactions.

function [Reta1, Reta2] = REA3(eta1,eta2,c,p)
N=length(eta1);
nA =zeros(N,1);
nI =zeros(N,1);
Reta1=zeros(N,3);
```

```
Reta2=zeros(N,3);
Vone = [1 1 1];
Done = [0 0 0];

for r = 2:N

  nA(r) =eta1(r,1)+eta1(r,2)+eta1(r,3)+eta1(r-1,1)+eta1(r-1,2)+eta1(r-1,3);
  nI(r) =eta2(r,1)+eta2(r,2)+eta2(r,3)+eta2(r-1,1)+eta2(r-1,2)+eta2(r-1,3);
end
  nA(1) = eta1(N,1)+eta1(N,2)+eta1(N,3)+eta1(1,1)+eta1(1,2)+eta1(1,3);
  nI(1) = eta2(N,1)+eta2(N,2)+eta2(N,3)+eta2(1,1)+eta2(1,2)+eta2(1,3);
Diff = nA - c*nI;

  for r = 1:N

      if Diff(r)>0 & (rand>(1-p(1)))
           Reta1(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(2)))
           Reta1(r,:) = Done(:);
      else
          Reta1(r,:) = eta1(r,:);
      end

        if Diff(r)>0 & (rand>(1-p(3)))
           Reta2(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(4)))
           Reta2(r,:) = Done(:);
      else
          Reta2(r,:) = eta2(r,:);
        end
   end
```

6.2.6 Reaction Rule 4

```
%1D Reaction rule (4.5) and (4.6) for interaction neighborhood N4={r-1,r,r+1}.
%Input: 1. The states of all cells of species A and I before reactions.
%       2. Probability p (vector) and a constant c which control the reactions.
%Output: The states of all cells of species A and species I after reactions.

function [Reta1, Reta2] = REA4(eta1,eta2,c,p)
N=length(eta1);
nA =zeros(N,1);
nI =zeros(N,1);
```

```
Reta1=zeros(N,3);
Reta2=zeros(N,3);
Vone = [1 1 1];
Oone = [0 0 0];

for r = 2:N-1
 nA(r) = eta1(r-1,1)+eta1(r-1,2)+eta1(r-1,3)+eta1(r,1)+eta1(r,2)+...
         +eta1(r,3)+eta1(r+1,1)+eta1(r+1,2)+eta1(r+1,3);
 nI(r) = eta2(r-1,1)+eta2(r-1,2)+eta2(r-1,3)+eta2(r,1)+eta2(r,2)+...
         +eta2(r,3)+eta2(r+1,1)+eta2(r+1,2)+eta2(r+1,3);
end
  nA(1) = eta1(N,1)+eta1(N,2)+eta1(N,3)+eta1(1,1)+eta1(1,2)+eta1(1,3)+...
          +eta1(2,1)+eta1(2,2)+eta1(2,3);
  nA(N) = eta1(N,1)+eta1(N,2)+eta1(N,3)+eta1(N-1,1)+eta1(N-1,2)+eta1(N-1,3)+...
          +eta1(1,1)+eta1(1,2)+eta1(1,3);
  nI(1) = eta2(N,1)+eta2(N,2)+eta2(N,3)+eta2(1,1)+eta2(1,2)+eta2(1,3)+...
          +eta2(2,1)+eta2(2,2)+eta2(2,3);
  nI(N) = eta2(N,1)+eta2(N,2)+eta2(N,3)+eta2(N-1,1)+eta2(N-1,2)+...
          +eta2(N-1,3)+eta2(1,1)+eta2(1,2)+eta2(1,3);

  Diff = nA - c*nI;
  for r = 1:N

      if Diff(r)>0 & (rand>(1-p(1)))
           Reta1(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(2)))
           Reta1(r,:) = Oone(:);
      else
          Reta1(r,:) = eta1(r,:);
      end

        if Diff(r)>0 & (rand>(1-p(3)))
           Reta2(r,:) = Vone(:);
      elseif Diff(r) <0 &  (rand>(1-p(4)))
           Reta2(r,:) = Oone(:);
      else
          Reta2(r,:) = eta2(r,:);
        end
    end
```

6.2.7 1D LGCA Driver

```
clear;clc;close all
```

```
%Initialization
L = 100;                %Length of lattices L_a = L_i
Nt=300;                 %Number of time steps
N = L;                  %Number of cells
m_A=1;m_I=7;            %Speeds of Activator A and Inhibitor I
c=1;
p=[1 1 1 1];
etaA=zeros(N,3);
etaI=zeros(N,3);
etaA(50,3)=1;           %Initial condition
for t=1:Nt
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %1. Reaction Process
     [RetaA,RetaI]=REC1(etaA,etaI,c,p);
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %2. Shuffling Process
     MetaA = SHUFF(RetaA);
     MetaI = SHUFF(RetaI);
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %3. Propagation Process
     etaA = PROP(MetaA,m_A);
     etaI = PROP(MetaI,m_I);
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %4. Concentration of species A
     solA(:,t) = etaA(:,1)+etaA(:,2)+etaA(:,3);
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end

  %5. Plot the evolution of the Activator A
imagesc(solA)
title(['Evolution of species A at time=',num2str(t)])
colormap([1 1 1; .8 .8 .8; .5 .5 .5; 0 0 0; 1 0 1]);
xlabel('Time')
ylabel('Space')
```

6.3 2D LGCA Codes

In this section, the matlab codes for the 2D LGCA (with four velocity channels and one rest channel) are attached.

6.3.1 2D Propagation Rule

```
%Propagation process
```

```
function [PA] = PROP2D(A,m)
SIZE= size(A);
N=SIZE(2);
channel=SIZE(1);
PA = zeros(channel,N,N);
%Horizontal propagation
for j = 1:N
    PA(1,1+m:N,j) =   A(1,1:N-m,j);
    PA(1,1:m,  j) =   A(1,N-m+1,j);
    PA(3,1:N-m,j)   = A(3,m+1:N,j);
    PA(3,N-m+1:N,j) = A(3,1:m,j);

end
%Vertical propapation
for i = 1:N
    PA(2,i,1+m:N) =   A(2,i,1:N-m);
    PA(2,i,1:m)   =   A(2,i,N-m+1:N);
    PA(4,i,1:N-m) =   A(4,i,m+1:N);
    PA(4,i,N-m+1:N) = A(4,i,1:m);
end
%Rest channel
for j= 1:N
    PA(5,1:N,j) = A(5,1:N,j);
end
```

6.3.2 2D Shuffling Rule

```
% Shuffling process
function [M]=SHUFF2D(R)
SIZE= size(R);
N=SIZE(2);
channel=SIZE(1);
M = zeros(channel,N,N);
for i = 1:N
    for j = 1:N
       temp = randperm(channel);
       P=zeros(channel,channel);
           for jj = 1:channel
               P(jj,temp(jj)) = 1;
           end
        M(:,i,j) = P*R(:,i,j);
    end
end
```

6.3.3 2D Reaction Rule 1

```
%2D Reaction rule (4.5) and (4.6) interaction neighborhood N1={i,j}.
%Input: 1. The states of all cells of species A and B before reactions.
%       2. Probability p (vector) and a constant c which control the reactions.
%Output: The states of all cells of species A and species B after reactions.
function [RA,RB]=REA2DN1(A,B,p,c)
SIZE= size(A);
N=SIZE(2);
channel=SIZE(1);
RA = zeros(channel,N,N);
RB = zeros(channel,N,N);
nA=RA;nB=nA;Diff=nA-nB;
Vone = [1 1 1 1 1];
Oone = [0 0 0 0 0];
for i = 1:N
    for j = 1:N
 nA(i,j) = sum(A(:,i,j));
 nB(i,j) = sum(B(:,i,j));
 Diff(i,j) = nA(i,j) -c*nB(i,j);
    end
end
for  i = 1:N
    for j=1:N
        if Diff(i,j) >0 & (rand>(1-p(1)))
            RA(:,i,j) = Vone;
        elseif Diff(i,j) <0 & (rand>(1-p(2)))
            RA(:,i,j) = Oone;
        else
             RA(:,i,j) = A(:,i,j);
        end

        if Diff(i,j) >0 & (rand>(1-p(3)))
            RB(:,i,j) = Vone;
        elseif Diff(i,j) <0 & (rand>(1-p(4)))
            RB(:,i,j) = Oone;
        else
             RB(:,i,j) = B(:,i,j);
        end

    end
end
```

6.3.4 2D Reaction Rule 2

```
%2D Reaction rule (4.5) and (4.6)
```

```
%Interaction neighborhood N2= ({i,j},{i-1,j},{i+1,j},{i,j-1},{i,j+1}).
%Input: 1. The states of all cells of species A and B before reactions.
%       2. Probability p (vector) and a constant c which control the reactions.
%Output: The states of all cells of species A and species B after reactions.
function [RA,RB]=REA2DN2(A,B,p,c)
SIZE= size(A);
N=SIZE(2);
channel=SIZE(1);
RA = zeros(channel,N,N);
RB = zeros(channel,N,N);
nA=RA;nB=nA;Diff=nA-nB;
Vone = [1 1 1 1 1];
Oone = [0 0 0 0 0];
for i = 2:N-1
     for j = 2:N-1
  nA(i,j) = sum(A(:,i,j))+sum(A(:,i,j+1))+sum(A(:,i,j-1))+sum(A(:,i-1,j))+sum(A(:,i+1,j));
  nB(i,j) = sum(B(:,i,j))+sum(B(:,i,j+1))+sum(B(:,i,j-1))+sum(B(:,i-1,j))+sum(B(:,i+1,j));
     end
end
 nA(1,1) = sum(A(:,1,1))+ sum(A(:,1,2))+sum(A(:,2,1))+sum(A(:,1,N))+sum(A(:,N,1));
 nA(1,N) = sum(A(:,1,N))+ sum(A(:,1,N-1))+sum(A(:,1,1))+sum(A(:,2,N))+sum(A(:,N,N));
 nA(N,1) = sum(A(:,N,1))+ sum(A(:,N,2))+sum(A(:,N,N))+sum(A(:,N-1,1))+sum(A(:,1,1));
 nA(N,N) = sum(A(:,N,N))+ sum(A(:,N,1))+sum(A(:,N,N-1))+sum(A(:,N-1,N))+sum(A(:,1,N));
 nB(1,1) = sum(B(:,1,1))+ sum(B(:,1,2))+sum(B(:,2,1))+sum(B(:,1,N))+sum(B(:,N,1));
 nB(1,N) = sum(B(:,1,N))+ sum(B(:,1,N-1))+sum(B(:,1,1))+sum(B(:,2,N))+sum(B(:,N,N));
 nB(N,1) = sum(B(:,N,1))+ sum(B(:,N,2))+sum(B(:,N,N))+sum(B(:,N-1,1))+sum(B(:,1,1));
 nB(N,N) = sum(B(:,N,N))+ sum(B(:,N,1))+sum(B(:,N,N-1))+sum(B(:,N-1,N))+sum(B(:,1,N));
 for j=2:N-1
    nA(1,j) =sum(A(:,1,j))+ sum(A(:,1,j+1))+sum(A(:,1,j-1)) + sum(A(:,N,j))+sum(A(:,2,j)) ;
    nA(N,j) =sum(A(:,N,j))+ sum(A(:,N,j+1))+sum(A(:,N,j-1)) +sum(A(:,N-1,j))+sum(A(:,1,j));
     nB(1,j) =sum(B(:,1,j))+ sum(B(:,1,j+1))+sum(B(:,1,j-1)) + sum(B(:,N,j))+sum(B(:,2,j)) ;
    nB(N,j) =sum(B(:,N,j))+ sum(B(:,N,j+1))+sum(B(:,N,j-1)) +sum(B(:,N-1,j))+sum(B(:,1,j));
 end
 for i=2:N-1
     nA(i,1)= sum(A(:,i,1))+ sum(A(:,i,2))+sum(A(:,i,N))+   sum(A(:,i-1,1))+sum(A(:,i+1,1));
     nA(i,N)= sum(A(:,i,N))+ sum(A(:,i,1))+sum(A(:,i,N-1))+ sum(A(:,i-1,N))+sum(A(:,i+1,N));
     nB(i,1)= sum(B(:,i,1))+ sum(B(:,i,2))+sum(B(:,i,N))+   sum(B(:,i-1,1))+sum(B(:,i+1,1));
     nB(i,N)= sum(B(:,i,N))+ sum(B(:,i,1))+sum(B(:,i,N-1))+ sum(B(:,i-1,N))+sum(B(:,i+1,N));
 end
 Diff = nA-c*nB;
for  i = 1:N
    for j=1:N
        if Diff(i,j) >0 & (rand>(1-p(1)))
            RA(:,i,j) = Vone;
        elseif Diff(i,j) <0 & (rand>(1-p(2)))
            RA(:,i,j) = Oone;
        else
```

```
            RA(:,i,j) = A(:,i,j);
        end

        if Diff(i,j) >0 & (rand>(1-p(3)))
            RB(:,i,j) = Vone;
        elseif Diff(i,j) <0 & (rand>(1-p(4)))
            RB(:,i,j) = Done;
        else
            RB(:,i,j) = B(:,i,j);
        end

    end
end
```

6.3.5 2D LGCA Driver

```
% 2D 2-Component Lattice Gas CA for Activator-Inhibitor System based on the
% modified reaction rule (4.5) and (4.6).
% A: Activator
% B: Inhibitor
% Interacation neighborhood N = N1 = {i,j}.
% s= 5; Four velocity channels and one rest channel.
clear; close all; clc

% Initialization
m_A=1; m_B=11;p = [1 1 1 1 ];c=1;% Parameters from local interaction rules.
N = 100; % size of lattice
Nt= 500; % # of time steps
channel = 5; % # of channels on each node r = (i,j).
             % 4 velocity channels (C1,C2,C3,C4) and 1 rest channel C5
A = zeros(channel,N,N);
B = zeros(channel,N,N);
RA=A; RB=B;MA=A; MB=B;
% Initial condition
A(5,50,50)=1;
nA = zeros(N,N);%Concentration of species A
nB = nA;        %Concentration of species B
Diff = nA-nB;
solA = zeros(N,N);
iter =0;

for t =1:Nt

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %1. Reaction process
        [RA,RB] = REA2DN2(A,B,p,c);
```

```
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %2. Shuffling process
        MA = SHUFF2D(RA);
        MB = SHUFF2D(RB);
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %3. Propagation process
        A = PROP2D(MA,m_A);
        B = PROP2D(MB,m_B);
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %4. Compute and plot the concentration of species A
    if mod(iter,10)== 0
        t
      for i = 1:N
         for j = 1:N
           solA(i,j) = sum(A(:,i,j));
         end
       end
      imagesc(solA)
      colormap([1 1 1; .8 .8 .8; .6 .6 .6; .4 .4 .4; .2 .2 .2; 0 0 0; 1 0 1]);
      pause(.2)
    end
    iter = iter+1;

end
 xlabel('\bf x')
    ylabel('\bf y')
   Title(['Concentration of the activator A at time Nt =', num2str(t)])
```

Bibliography

[1] Deutsch, A., Dormann, S., (2004) Cellular Automaton Modeling of Biological Pattern Formation: Characterization, Application and Analysis. *Birkhauser.*

[2] Wolfram, S. (2002) A New Kind of Science. *Wolfram Media, Inc.*, ISBN 1-57955-008-8.

[3] Wittenberg, R. (2005). APMA990 class notes.

[4] Murray, J.D. (2002). Mathematical Biology. *Springer-Verlag.*

[5] Turing, A.M. (1952). The Chemical Basis of Morphogenesis. *Phil. Trans. Royal Society*, vol. B237, pp. 37 - 72.

[6] Yang, L.F., Zhabotinsky, A. M., Epstein, I.R. (2004). Stable Squares and Other Oscillatory Turing Patterns in a Reaction-Diffusion Model. *Physical Review Letters* vol. 92, 198308.

[7] Boon, J.P., et al. (1996). Reactive Lattice Gas Cellular Automata. *Physics Reports 273*, 55-147.

[8] Weimar, J.R. (1997). Cellular Automata for Reaction-Diffusion Systems. *Parallel Computing* 1699-1715.

[9] Ruuth, S.J. (1995). Implicit-explicit methods for reaction-diffusion problems. *J. Math. Biology*, 34:148-176.

[10] Lawniczak, A. T. (1997). Lattice gas automata for diffusive-convective transport dynamics. *Center for Nonlinear Studies, Newsletter No. 136*, LALP-97-010 B237, pp. 37 - 72.

[11] Nishiura, Y., Ueyama, D. (1999). A skeleton structure of self-replicating dynamics. *Physica D*, vol. 130, 73-104.

[12] Nishiura, Y., Ueyama, D. (2000). Self-Replication, Self-Destruction, and Spatio-Temporal Chaos in the Gray-Scott Model, *Forma*, 15, 281-289.