UNIVERSITY OF CINCINNATI

Date: 18-Nov-2009

I, Balaji Ananthanpillai

hereby submit this original work as part of the requirements for the degree of:

Master of Science

in Electrical Engineering

It is entitled:

Stochastic Simulation of the Phage Lambda System and the Bioluminescence System Using the Next Reaction Method

Student Signature: Balaji Ananthanpillai

This work and its defense approved by:

Committee Chair: Carla Purdy, C, PhD  
Carla Purdy, C, PhD

Wen-Ben Jone, PhD  
Wen-Ben Jone, PhD

George Purdy, PhD  
George Purdy, PhD
Stochastic Simulation of the Phage Lambda System and the Bioluminescence System Using the Next Reaction Method

A thesis submitted to the

Graduate school
of the University of Cincinnati
in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in the Department of Electrical and Computer Engineering
of the College of Engineering
University of Cincinnati

November, 2009

by

Balaji Ananthanpillai
B.Tech (EEE), Jawaharlal Nehru Technological University, India 2006

Thesis Advisor and Committee Chair: Dr. Carla Purdy
Abstract

A biological pathway represents a network of complex reactions at the molecular level of living cells. Pathways model how biological molecules interact to carry out a biological function and how they respond to external stimuli. The pathway models are derived through scientific experimentation and data analysis. Modeling biochemical reactions is a component in studying and analyzing biological pathways. The modeling and control of biological pathways is extremely important in understanding various biological phenomena. Biological pathways can be controlled in the laboratory and this process has applications in medicine and also in the invention of certain biologically based components for computer circuits or bio-circuits. However, direct experimental study of the pathways is expensive and also time consuming. Mathematical modeling and stochastic simulation techniques provide an affordable and easy to use experimental platform. Mathematical modeling and stochastic simulations, based on biochemical rate equations, provide us with rigorous tools to understand the intricacies involved in biological pathways. In this thesis we model the Phage Lambda system, which functions as a biological switch, and the Bioluminescence system, which provides a model of communication. We use the stochastic simulation techniques based on Gibson and Bruck’s Next Reaction Method. We compare our results with the results of simulations using Gillespie’s Algorithm and with the results from an agent-based model. We also provide easy-to-use templates for the simulations to support future research in this area.
Acknowledgements

I would like to thank my advisor Dr. Carla Purdy for her continuous and valuable guidance and support all through my thesis work. Her valuable suggestions and insights helped several times to circumvent the problems that I had to face, during the tough paths of my research. This work would not have been possible if not for her continuous encouragement and mentorship. I also would like to thank Dr. Wen Ben Jone and Dr. George Purdy for agreeing to be a part of my defense committee and for giving some valuable inputs to improve the quality of work.

This thesis is dedicated to my wonderful parents who have been a constant source of inspiration, strength and support to me. If it were not for them, I would have never had the opportunity to do my Masters at the US. I would like to make a special mention of my mom without whom I would not have been able to carry out my research at UC. I am highly indebted to her for the sleepless night she spent praying for my welfare.

This part of acknowledgments would make no sense without mentioning about my wonderful and best friends, Arpita and Soumya. Life at UC would never have been the same without them. Their continuous encouragement and motivation does deserve a special mention without which I would never have been able to carry out my research properly. I would also like to make a special mention of my Ramani periamma who ensured that my spirits were high all the while. Above all, I would like to dedicate this thesis to my late grandmother who had been my source of inspiration all through my life. Finally, I would like to thank God for his wonderful blessings and for giving me the strength and courage to live up to the expectations of my family and friends.
# Contents

1 Introduction to Biological Pathways ............................................................................. 1
   1.1 Introduction ........................................................................................................ 1
   1.2 Motivation .......................................................................................................... 3
   1.3 Objective ........................................................................................................... 4
   1.4 Approach ........................................................................................................... 5
   1.5 Thesis outline .................................................................................................... 5

2 Mathematical and Stochastic Simulation Methods ....................................................... 7
   2.1 Simulation methods for biochemical systems ................................................. 7
   2.2 Importance of modeling and simulation methods ........................................... 7
   2.3 Tools for simulation ......................................................................................... 8
      2.3.1 MATLAB and its applications ................................................................. 8
         2.3.1.1 Ordinary differential equation models available in Matlab ............... 9
      2.3.2 Using ODE’s for simulating biomolecular systems ................................. 16
   2.4 Occurrence of stochasticity in biological systems ........................................... 18
   2.5 Stochastic simulation ....................................................................................... 19
      2.5.1.1 Advantages of stochastic simulation .................................................... 20
      2.5.1.2 Disadvantages of stochastic simulation .............................................. 20
      2.5.2 Stochastic models ...................................................................................... 20
2.5.2.1 Gillespie Algorithm ................................................................. 21
2.5.2.2 Next Reaction Method ............................................................. 22
2.6 Other stochastic simulation methods .............................................. 24
2.7 Agent Based Modeling (ABM) ......................................................... 25
  2.7.1 Advantages of Agent Based Modeling (ABM) ......................... 27
  2.7.2 Drawbacks of ABM ................................................................. 27
2.8 Limitations of biological model simulation ...................................... 27

3 Modeling the Phage Lambda System ................................................. 29
  3.1 Introduction to biological pathways ............................................. 29
  3.2 Phage Lambda system ................................................................. 30
  3.3 Modeling the Phage Lambda system ............................................. 32
  3.4 Differential equations modeling the Phage Lambda system .......... 34
  3.5 Mathematical simulation of the Phage Lambda system ................ 39
  3.6 Stochastic method implementation for the Phage Lambda system .... 40

4 Modeling the Bioluminescence System ............................................. 42
  4.1 Bioluminescence system ............................................................. 42
    4.1.1 Application of the Bioluminescence system ......................... 44
    4.1.2 Model-I of the Bioluminescence system .............................. 44
    4.1.3 Model-II of the Bioluminescence system ......................... 46
    4.1.4 Model used in this research ............................................. 47
4.2 Differential equations modeling the *V. Fischeri* system ........................................ 48
  4.2.1 Mathematical simulation of the Bioluminescence system ........................................ 57
  4.2.2 Stochastic modeling of the Bioluminescence system ........................................ 59
  4.3 Gillespie’s Algorithm for stochastic modeling ......................................................... 59
  4.4 Gibson and Bruck’s Next Reaction Method .............................................................. 61
    4.4.1 Dependency graph ................................................................................................. 63
    4.4.2 Indexed priority queue .......................................................................................... 64

5 Experiments and Results .................................................................................................. 65
  5.1 Deterministic simulation of the Phage Lambda system .................................................. 65
  5.2 Implementation of the stochastic simulation method ..................................................... 73
    5.2.1 Python programming language .............................................................................. 74
    5.2.2 Generalized template for the Python code used in the research ......................... 76
  5.3 Stochastic simulation of the Phage Lambda system ..................................................... 80
  5.4 Deterministic simulation of the Bioluminescence system ........................................... 84
  5.5 Stochastic simulation of the Bioluminescence system ................................................. 90

6 Conclusions and Future Work .......................................................................................... 95
  6.1 Introduction .................................................................................................................. 95
  6.2 Conclusions ................................................................................................................ 95
  6.3 Suggestions for future work ....................................................................................... 96

References: .......................................................................................................................... 98
List of Figures

Figure 2.1 The pink region (disk) shows the stability region for Euler method. Reprinted from [27]................................................................. 13

Figure 2.2 The pink region represents the stability region for the Trapezoidal Method. Reprinted from [27].................................................................................................................. 14

Figure 2.3 The pink region represents the stability for the second-order Adams-Bashforth method. Reprinted from [27]........................................................................................................ 15

Figure 3.1 Graphical Representation of chemotaxis pathways in E.coli bacteria. Reprinted from [54] .................................................................................................................. 29

Figure 3.2 Lysis plaques of Lambda Phage in E.coli bacteria [57] ................................................................. 31

Figure 3.3 Analogy between an electrical inverter and the phage lambda switching states. Reprinted from [6] ........................................................................................................ 32

Figure 4.1 Structure of a gram negative cell wall. Reprinted from [62] ...................................................... 42

Figure 4.2 Basic model of the V.Fischeri system. Reprinted from [9] .............................................................. 43

Figure 4.3 Model-I depicting the Bioluminescence process. Reprinted from [9] ............................................ 46

Figure 4.4 Model-II describing the Bioluminescence process. Reprinted from [9] ....................................... 47

Figure 4.5 One step simulation flow of the Gillespie Algorithm and the Gibson and Bruck’s Next Reaction Method. Reprinted from [46] .................................................................................. 63

Figure 4.6 Dependency graph of the Phage Lambda system................................................................. 64

Figure 5.1 Transient characteristics of the phage lambda inverter using ode45 solver (Matlab) ......... 66

Figure 5.2 Transient characteristics of the Phage Lambda system in the time range [0 500] using ode45 solver (Matlab) ........................................................................................................ 67
Figure 5.3 Transient characteristics of the Phage Lambda system in the time range [500 1000] using ode45 solver (Matlab) ................................................................. 68
Figure 5.4 Transient characteristics of the Phage Lambda system in the time range [1000 1500] using ode45 solver (Matlab) ................................................................. 68
Figure 5.5 Transient characteristics of the Phage Lambda system in the time range [1500 2000] using ode45 solver (Matlab) ................................................................. 69
Figure 5.6 Transient characteristics of the Phage Lambda system using ode113 solver (Matlab) ................................................................. 70
Figure 5.7 Transient characteristics of the Phage Lambda system in the time range [0 500] using ode113 solver (Matlab) ................................................................. 70
Figure 5.8 Transient characteristics of the Phage Lambda system in the time range [500 1000] using ode113 solver (Matlab) ................................................................. 71
Figure 5.9 Transient characteristics of the Phage Lambda system in the time range [1000 1500] using ode113 solver (Matlab) ................................................................. 71
Figure 5.10 Transient characteristics of the Phage Lambda system in the time range [1500 2000] using ode113 solver (Matlab) ................................................................. 72
Figure 5.11 Transient characteristics of the Phage Lambda system using ode15s solver (Matlab). Reprinted from [9]. ................................................................. 73
Figure 5.12 Screenshot showing the gillespie_helper.py template ................................................................. 77
Figure 5.13 Screenshot showing the nextrxn.py template which describes Gibson and Bruck’s Next Reaction Method ................................................................. 78
Figure 5.14 Screenshot of the command window from which the Next Reaction Method is executed ................................................................. 78
Figure 5.15 Screenshot of the text file describing the reactions of the Phage Lambda system.... 79
Figure 5.16 Screenshot of the text file showing the initial number of molecules of the Phage Lambda system ........................................................................................................................................... 79

Figure 5.18 Stochastic simulation of the “ON” state or the Lytic State of the Phage Lambda System .................................................................................................................................................. 81

Figure 5.19 Stochastic simulation of the Lysogenic or the “OFF” state of the Phage Lambda System .................................................................................................................................................. 82

Figure 5.20 Stochastic simulation of the “ON” state or the Lytic state of the Phage Lambda system using the Gillespie Algorithm. Reprinted from [9] ................................................................................................................................. 82

Figure 5.21 Stochastic simulation of the “OFF” state or the Lysogenic state of the Phage Lambda system using the Gillespie Algorithm. Reprinted from [9] ................................................................................................................................. 83

Figure 5.22 Simulation of the Phage Lambda system using Agent Based Modeling. Reprinted from [13] .................................................................................................................................................. 83

Figure 5.23 Simulation results of the bioluminescence system using the Matlab ode45 solver with the input $A_{\text{source}} = 0$ ........................................................................................................................................... 86

Figure 5.24 Simulation results of the bioluminescence system using the Matlab ode45 solver with the input $A_{\text{source}} = 10$ ........................................................................................................................................... 86

Figure 5.25 Simulation of the Bioluminescence system using the Matlab ode113 solver with the input $A_{\text{source}} = 0$ ........................................................................................................................................... 88

Figure 5.26 Simulation of the Bioluminescence system using the Matlab ode113 solver with input $A_{\text{source}} = 10$ ........................................................................................................................................... 88

Figure 5.27 Simulation results of the Bioluminescence system using the ode15s solver with the input $A_{\text{source}} = 0$. Reprinted from [9] ........................................................................................................................................... 89
Figure 5.28 Simulation results of the Bioluminescence system using the ode15s solver with the input $A_{\text{source}} = 10$. Reprinted from [9]. ............................................................... 89

Figure 5.29 Stochastic simulation of the bioluminescence system using the Next Reaction method showing the input $A_i$ and output LuxA/B. LuxR and LuxI are also included. .............. 92

Figure 5.30 Stochastic simulation of the bioluminescence system using the Gillespie Algorithm showing the input $A_i$ and output LuxA/B. LuxR and LuxI are also included [9]....................... 92

Figure 5.31 Figure showing the luminescence response of a cell in a colony [5]. ......................... 93

List of Tables

Table 2.1 Description of different type of Matlab ODE Solvers [23][24][25].......................... 10

Table 2.2 Comparison of different stochastic algorithms [4] ..................................................... 24

Table 3.1 Rate constants for the Phage Lambda system..........................................................40

Table 4.1 Biochemical species and the initial concentrations of the ODE model of the bioluminescence system [9]...........................................................58

Table 4.2 Rate constants for the bioluminescence system [9]..............................................58

Table 5.1 Rate constants for modeling the Phage Lambda system [9].................................65

Table 5.2 Initial number of molecules for the biochemical species for the “ON” or Lytic state . 81

Table 5.3 Initial number of molecules for the biochemical species for the “OFF” or Lysogenic state ...................................................................................................................... 81

Table 5.4 Biochemical species and the initial concentrations of the ODE model of the bioluminescence system [9].................................................................................84

Table 5.5 Rate constants modeling the bioluminescence system [9].....................................85

Table 5.6 Initial values (number of molecules) of the biochemical species for the stochastic model of the Bioluminescence system.........................................................91
1 Introduction to Biological Pathways

1.1 Introduction

In this chapter we introduce biological pathways, mathematical and stochastic modeling and the motivation behind this current research. The objective and goals of the research are clearly explained. An outline for the remaining chapters of the thesis is neatly framed and stated.

A biological pathway is a representation of a network of complex reactions at the molecular level of living cells. These pathways model how the biological molecules interact to achieve a biological function and also how they respond to external stimuli. The pathways are derived through scientific experimentation and data analysis and these pathways also capture the current knowledge of the biological processes. Generally, pathways are represented as graphs, consisting of nodes and edges. A node in a biological pathway represents a biological molecule but it can also be used to summarize a totally different pathway that interacts with the current pathway under study or it might even be used to represent some relevant phenomenon such as an external environmental stimulus (e.g., heat or light). A node which represents a biological molecule in a pathway diagram may be either a metabolite or a nucleic acid or a protein. DNA, mRNA, tRNA and structural RNA are all nucleic acids. Enzymes, structural proteins, chemical effectors etc. can be proteins. An edge in a pathway represents some sort of interaction or relationship between the nodes. This interaction between the nodes could be of many types such as gene expression, inhibition, catalysis, chemical modification etc. [1]. Some actions in a biological pathway might lead to a certain product or even a change in the cell. Such a pathway paves the way for the assembly of new molecules, such as fat or protein. Researchers have
observed that biological pathways are far more complex than early experiments thought. Most biological pathways do not start at some point ‘X’ and end at a point ‘Y’. Biological pathways have been discovered through laboratory studies of cultured cells, flies, mice and several other organisms. As already discussed, a biological pathway is represented as a graph that consists of nodes and edges [2]. The biological pathways discussed in this thesis are the Phage Lambda system and the Bioluminescence system. The modeling and control of biological pathways is extremely important to study the various biological phenomena. Biological pathways can be controlled in the laboratory and this process has applications in medicine and also in the invention of certain biologically based components for computer circuits or bio-circuits. However, this direct experimental study of the pathways is expensive and also time consuming. Mathematical modeling and stochastic simulations, based on biochemical rate equations, provide us with certain rigorous tools to resolve the complexities involved in the biological pathways [3]. The mathematical modeling is a very powerful approach that has been developed for understanding the complexities of the biological systems. There are different complex biological processes like metabolic pathways, gene regulatory networks and cell signaling pathways, and of late several successful attempts have been made to model and simulate these complex pathways. The pathway models have provided us with some valuable insight into the behavior of complex biological systems [4].

The study of these pathways is one area in systems biology. Systems biology is an innovative field where biological pathways, genetic structures and protein production are analyzed using mathematical models, simulations and statistical techniques. Systems biology actually refers to a biology-based inter-disciplinary field that primarily focuses on systematic study of complex interactions occurring in biological systems by using a new perspective or method [5]. In other
words, systems biology studies the various interactions between the components that occur in a system and how these interactions give rise to the behavior of a system as a whole. A good understanding of biological pathways and genetics contributes to the improving of human health and also in managing natural and human resources. In order to make significant contributions to this area of research, one must have knowledge of diverse fields which include not only biology, but also chemistry, mathematics, electrical and computer engineering and also computer science.

1.2 Motivation

The main motivation behind this research is to provide better computational support for biologists to carry out various complex biological experiments “in silico” (performed on computer or computer simulation) and also to understand the usage of bio-components for computation. It is very difficult to understand the complex behavior of cellular networks because of the complexity of the pathway interactions and also because of the large number of components involved. In order to understand the spatial properties and dynamics of such complex networks, several mathematical models and computer simulation techniques have been developed. Biology in silico has an edge over conventional experimental biology in terms of cost, ease and speed. Modeling can provide certain valuable insights into the working and the organization of biological systems. It can also deal with random events which form an important aspect of modeling cellular networks.

A second motivation is to study the biochemical circuit components. Customarily VLSI circuits are built using CMOS transistors and these circuits are fabricated on silicon. But in recent times there has been a considerable decrease or shrink in the size of the transistors and hence scientists are looking for alternate technologies that can be used to develop the next generation circuits. The two important areas in this direction of study are nano-devices and bio-circuits [6]. In a
CMOS transistor, the magnitude of stochastic effects is quite small and can be ignored. However, this cannot be the case while building the nano-circuits and bio-circuits.

The current research in systems biology can be broadly divided into two areas. In the first area the stochasticity of the biological systems is studied. In the second area the modeling approaches, certain mathematical formalisms and simulation algorithms are discussed [4].

1.3 Objective

The main goals of the current research are to further study the optimization and control of the biological pathways, namely the Phage Lambda system and the Bioluminescence system. This involves changing the model parameters such as the rate constants and the species concentration to achieve the desired levels of the output proteins within a certain specified time interval. Normally, control of these biological systems is achieved by some random lab experiments [7] or certain directed evolution [8]. The lab experiments and directed evolutions are supplemented with mathematical modeling and stochastic simulations. Therefore, the major objective of our research is the optimization and control of the biological systems using mathematical modeling and stochastic simulations. In the subsequent chapters, the mathematical models and the stochastic simulations that are used in this research are discussed in detail. R. Krishnan [9] had already carried out the stochastic simulation for the above mentioned biological pathways based on Daniel T. Gillespie’s method [10] which is popularly known as the Gillespie Algorithm. The current research involves updating the stochastic simulation method to be used to control the biological pathways. This method gives higher accuracy and faster simulation time.
1.4 Approach

A simulation method can usually be either deterministic or stochastic. A deterministic model is a kind of mathematical model in which every variable changes according to a certain mathematical formula but not by some random fluctuations whereas a stochastic model is a kind of mathematical model where there is room for random fluctuations in one or more of its variables. Therefore, in a stochastic model, one single estimate is not considered but a probability distribution of various possible estimates is taken into account. The deterministic model in this research is obtained by modeling the reactions using ordinary differential equations (ODE’s). An approximate solution to the reactions can be obtained by numerically integrating these ODE’s. The history behind the development of stochastic simulation algorithms and methods is to analyze the chemical reactions which involve a large number of species with complex reaction kinetics [11]. The very first stochastic simulation algorithm developed was the Gillespie Algorithm and was proposed by Daniel T. Gillespie in 1977 to simulate chemical and biochemical system of reactions effectively [10]. This algorithm is actually based on the Monte Carlo method. We will discuss in detail the Monte Carlo method and the Gillespie Algorithm in the subsequent chapters. The stochastic simulation method used in this research is the Next Reaction method by Gibson and Bruck which is an enhancement to Gillespie’s First Reaction Method [12].

1.5 Thesis outline

Chapter 2 gives an overview of different stochastic simulation methods, comparison of various stochastic simulation algorithms, the advantages of stochastic simulation methods over agent based modeling (ABM) [13] and its limitations.
**Chapter 3** provides a detailed description of the Phage Lambda system. This chapter also describes the reactions that have been modeled, the mathematical modeling and the stochastic simulation of the Phage Lambda system, the stochastic rules, spatial factors and assumptions.

**Chapter 4** provides a detailed description of the Bioluminescence system. This chapter also describes the reactions of the *V. Fischeri* system that have been modeled, the mathematical modeling and the stochastic simulation of the Bioluminescence system. This chapter also makes a brief mention of the Gillespie Algorithm used by R. Krishnan [9] and the Gibson and Bruck’s Next Reaction method [14] used for modeling the Phage Lambda system and the Bioluminescence system.

**Chapter 5** contains results obtained using the Next Reaction Method. The results include the graphs showing the switching mechanisms of the phage lambda system and its qualitative comparison against the ABM [13] results and the results obtained using the Gillespie Algorithm. It also includes the graphs showing the behavior of the bioluminescence system and its comparison with the results obtained from the Gillespie Algorithm.

**Chapter 6** provides conclusion to the work and also provides details about the areas that can be explored in the future.
2 Mathematical and Stochastic Simulation Methods

2.1 Simulation methods for biochemical systems

A deterministic model is a mathematical representation in which every variable changes according to a mathematical formula but there are no random fluctuations, whereas a stochastic model is a mathematical model which uses random variables in order to predict the probability distributions of potential outcomes [15], [16]. Our research primarily focuses on the stochastic simulation methods and modeling of biological pathways using the stochastic simulation techniques.

Chemically reacting systems have been conventionally simulated as deterministic systems by solving a set of coupled ordinary differential equations (ODEs). Though these conventional ODE-based approaches are sufficient for most systems, they fail to deal adequately with the stochasticity in certain biochemical systems formed by living cells [17].

2.2 Importance of modeling and simulation methods

The interactions in most biological pathway models are quite complex and involve a large number of components. Therefore it is almost impossible to completely understand the behavior of these networks. Mathematical modeling and computer simulation techniques play a major role in understanding the spatial properties and dynamics of such networks. The application of these techniques to biological systems has given rise to a new technique of in silico (performed on computer or through computer simulation) biology. Since in silico biology is performed through computer simulation, it has advantages over experimental biology in terms of speed, cost and ease of carrying out the experiment. Experimentation using a whole, living organism as
opposed to a partial or dead organism is called in vivo and experimentation performed in a controlled environment such as a test tube is called in vitro. Experiments that are infeasible in vivo or in vitro can be easily carried out in silico. The most important aspect of modeling biochemical networks is the occurrence of stochastic or random events [4]. In this chapter we will discuss in brief deterministic and stochastic simulation methods and the various aspects of stochasticity in biological systems.

2.3 Tools for simulation

In this section we will briefly describe the tools we will be using for simulation.

2.3.1 MATLAB and its applications

In this section we briefly describe the evolution of MATLAB and why and where we use it. MATLAB is essentially a numerical computing environment which was invented by Cleve Moler [18], in the late 1970’s. MATLAB has many useful features and can be used as an interactive mathematical piece of software that provides an interface to the users. MATLAB offers features like manipulating matrices, plotting of functions and data, implementation of certain algorithms, creation of some user interfaces and also interfacing with programs from several other languages. MATLAB also provides users with some unique features like graphical multidomain simulation and model-based design for certain dynamic and embedded systems through an additional package called “Simulink” [19]. MATLAB is written in the C programming language and it operates on multiple platforms. Since MATLAB has so many features and is compatible with so many programming languages and can operate on several platforms, we have implemented our thesis using MATLAB.
2.3.1.1 Ordinary differential equation models available in Matlab

Mathematical models play a vital role in modeling and characterizing biological systems [20]. In a mathematical form a complex biological system is usually characterized by a set of biochemical reactions which involve molecular species as reactants and products [21], [22]. Each of the biochemical reactions is governed by a rate constant and this rate constant quantifies the speed of the biochemical reaction. The rate constant is denoted by $k_x$, where ‘x’ indicates the name of the reaction. The chemical reaction network is developed as a mathematical model which attempts to describe the biological process accurately. These chemical reactions are then converted into a set of equations termed ordinary differential equations (ODE’s). An ordinary differential equation is an equation that involves a function and its derivatives with respect to time. An ordinary differential equation of order ‘n’ would be of the form

$$f(x, y, y', \ldots, y^{(n)}) = 0$$

Here, ‘$y$’ is a function of ‘$x$’, $y' = y^{(1)}$ is the first derivative of $y$ with respect to $x$ and $y^{(n)}$ is the $n^{th}$ derivative of $y$ with respect to $x$. This represents an ordinary differential equation (ODE). These ODE models can be solved by certain built in functions in MATLAB which are known as ODE solvers. Earlier there were only two ODE solvers in MATLAB, ODE23 and ODE45, but now we have several ODE solvers which are described in Table 2.1, along with their capabilities.
<table>
<thead>
<tr>
<th>Solver</th>
<th>Problem Type</th>
<th>Order of Accuracy</th>
<th>When to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ode45 (employs 4&lt;sup&gt;th&lt;/sup&gt; and 5&lt;sup&gt;th&lt;/sup&gt; order Runge-Kutta methods)</td>
<td>Non stiff</td>
<td>Medium</td>
<td>Should be used most of the times and should be the first solver to be tried.</td>
</tr>
<tr>
<td>ode23 (employs 2&lt;sup&gt;nd&lt;/sup&gt; and 3&lt;sup&gt;rd&lt;/sup&gt; order Runge-Kutta methods)</td>
<td>Non stiff</td>
<td>Low</td>
<td>Should be used for solving problems with certain crude error tolerances or moderately stiff problems.</td>
</tr>
<tr>
<td>ode113 (variable order Adams-Bashforth-Moulton PECE solver)</td>
<td>Non stiff</td>
<td>Low to High</td>
<td>Should be used for solving problems with constricted error tolerances or computationally intense problems.</td>
</tr>
<tr>
<td>ode15s (variable order solver based on Numerical Differential Formulas)</td>
<td>Stiff</td>
<td>Low to Medium</td>
<td>Should be used if the ODE45 is slow because of the stiffness of the problem.</td>
</tr>
<tr>
<td>ode23s (modified Rosenbrock 2&lt;sup&gt;nd&lt;/sup&gt; order formula)</td>
<td>Stiff</td>
<td>Low</td>
<td>Should be used if using crude error tolerances to solve stiff systems and mass matrix is constant.</td>
</tr>
<tr>
<td>ode23t (implementation of trapezoidal rule)</td>
<td>Moderately Stiff</td>
<td>Low</td>
<td>Should be used for moderately stiff problems if you need the solution without numerical damping.</td>
</tr>
<tr>
<td>ode23tb (an implicit Runge-Kutta formula with a first stage that is a trapezoidal rule step and a second stage that is a backward differentiation formula of order two)</td>
<td>Stiff</td>
<td>Low</td>
<td>Should be used if crude error tolerances are used to solve stiff systems.</td>
</tr>
</tbody>
</table>

Table 2.1 Description of different type of Matlab ODE Solvers [23][24][25].

In Table 2.1, by stiff system or stiff equation, we mean a differential equation for which there are some numerical methods which are numerically unstable. This could be avoided if the step size for the equation is made extremely small. The main idea behind the stiffness is that the equation includes certain terms which can lead to a rapid variation in the solution [26].
There are certain types of problems that can be characterized as stiff and they are:

a) Problems which are of the form

\[ \frac{dy}{dt} = a*y + f(t), \]  

where \( a \in \mathbb{C} \) and \(|a|\) is large. \( \mathbb{C} \) denotes the set of complex numbers.

b) Problems which are of the form

\[ \frac{dy}{dt} = A*y + f(t), \]  

where \( A \) is a square matrix and has at least one eigenvalue \( l \in \mathbb{C} \) and \(|l|\) is large.

c) Problems which are of the form

\[ \frac{dy}{dt} = f(y, t), \]  

where the jacobian of \( f \) has at least one eigenvalue \( m \in \mathbb{C} \) and \(|m|\) is large.

To understand the numerical methods such as Runge-Kutta methods, the Euler method, Trapezoidal method, Multistep method and the second-order Adams-Bashforth method used in MATLAB, we will apply them to the elementary differential equation \( \frac{dy}{dt} = a*y \) where \( a \in \mathbb{C} \) [27].

The solution of this equation is \( y(t) = k*e^{at} \). This solution approaches zero as \( t \to \infty \) and when \( Re(a) < 0 \). When a numerical method also exhibits the same behavior, then this particular method is considered to be A-stable [26]. The A-stable property is highly important because an A-stable method does not have instability problems.

The solution of the equation \( \frac{dy}{dt} = a*y \) takes the form \( y_{n+1} = \varphi(ha)y_n \), and by induction it takes the form \( y_n = \varphi(ha)^ny_0 \) when the Runge-Kutta method is applied to it, where the function \( \varphi \) is called the stability function. Therefore, the condition \( y_n \to 0 \) as \( n \to \infty \) is equivalent to the
condition \( |\varphi(ha)| < 1 \). Therefore, the region of absolute stability is the set \( \{ z \in \mathbb{C} \mid |\varphi(z)| < 1 \} \) and the Runge-Kutta method is A-stable in case the region of absolute stability contains the set \( \{ z \in \mathbb{C} \mid \text{Re}(z) < 0 \} \), which, in fact, is the left half plane.

In general, the stability function for the Runge-Kutta method with coefficients ‘\( A \)’ and ‘\( b \)’ is given as:

\[
\Phi(z) = \frac{\det(I - z^* A + z^* e^* b^T)}{\det(I - z^* A)},
\]

(2.5)

Where, ‘\( e \)’ represents the vector with ones. Since it is one polynomial divided by another, it is a rational function.

Now, we discuss the Euler method and its application to the test equation described above. When the Euler method is applied to the test equation, \( dy/dt = ay \), we obtain an equation of the form

\[
Y_{n+1} = y_n + h \cdot f(t_n,y_n) = y_n + h \cdot (a \cdot y_n) = y_n + h \cdot a \cdot y_n = (1 + h \cdot a) \cdot y_n
\]

(2.6)

Therefore, \( y_n = (1 + h \cdot a)^n \cdot y_0 \) and the stability function is given as \( \varphi(z) = 1 + z \). For this method, the region of absolute stability or the stability region is given by \( \{ z \in \mathbb{C} \mid |1+z| < 1 \} \) which is the region represented by the pink disk shown in Figure 2.1 below. Therefore, the Euler method is not A-stable.
The equation for the trapezoidal method is given by

\[ y_{n+1} = y_n + \frac{1}{2} \cdot h \cdot (f(t_n, y_n) + f(t_{n+1}, y_{n+1})) . \]  \hspace{1cm} (2.7)

This formula when applied to the test equation \( \frac{dy}{dt} = a \cdot y \) gives us the equation,

\[ y_{n+1} = y_n + \frac{1}{2} \cdot h \cdot (a \cdot y_n + a \cdot y_{n+1}) . \]  \hspace{1cm} (2.8)

On solving for \( y_{n+1} \), we get,

\[ y_{n+1} = \frac{(1 + \frac{1}{2} \cdot (h \cdot a)) / (1 - \frac{1}{2} \cdot (h \cdot a))}{y_n} . \]  \hspace{1cm} (2.9)

Therefore, the stability function is given by

\[ \varphi(z) = \frac{(1 + \frac{1}{2} \cdot z)}{(1 - \frac{1}{2} \cdot z)} \]  \hspace{1cm} (2.10)

and the region of absolute stability or the stability region is given by

\[ \{ z \in \mathbb{C} \mid |(1 + \frac{1}{2} \cdot z)/(1 - \frac{1}{2} \cdot z)| < 1 \} \]  \hspace{1cm} (2.11)

Since this region of absolute stability contains the left-half plane, the Trapezoidal method is A-stable.
Here, we describe the multistep methods in detail. The linear multistep method assumes the form,

\[ y_{n+1} = \sum_{i=0}^{s} l_i \cdot y_{n-i} + h \cdot \sum_{j=-1}^{s} m_j \cdot f(t_{n-j}, y_{n-j}). \]  
(2.12)

When this method is applied to the test equation, the above form changes to

\[ y_{n+1} = \sum_{i=0}^{s} l_i \cdot y_{n-i} + h \cdot a \cdot \sum_{j=-1}^{s} m_j \cdot y_{n-j} \]  
(2.13)

This can be further simplified to

\[ (1 - m_j \cdot z) \cdot y_{n+1} - \sum_{j=-1}^{s} (l_j + m_j \cdot z) \cdot y_{n-j} = 0 \]  
(2.14)

where, \( z = h \cdot a \). This is basically a recurrence relation and this method is considered to be A-stable if all the solutions \( \{y_n\} \) of the recurrence relation converges to zero when \( \text{Re}(z) < 0 \).

Now, we check to see if the Second-order Adams-Bashforth method is A-stable or not. The two-step Adams-Bashforth method is given by
\[ y_{n+1} = y_n + h \left( \frac{3}{2} f(t_n, y_n) - \frac{1}{2} f(t_{n-1}, y_{n-1}) \right) \]  

(2.15)

Then the characteristic polynomial is given by

\[
\Phi(w, z) = w^2 - (1 + \frac{3}{2} z) w + \frac{1}{2} z = 0
\]

(2.16)

and this characteristic polynomial has the roots,

\[
w = \frac{1}{2} \left( 1 + \frac{3}{2} z \pm \sqrt{(1 + z + \frac{9}{4} z^2)} \right)
\]

(2.17)

Therefore, the region of stability is given by

\[
\{ z \in \mathbb{C} | \left| \frac{1}{2} \left( 1 + \frac{3}{2} z \pm \sqrt{(1 + z + \frac{9}{4} z^2)} \right) \right| < 1 \}
\]

(2.18)

This region does not include the entire left-half plane but it only includes the portion of the real-axis between \( z = -1 \) and \( z = 0 \). Hence, the Adams-Bashforth method is not A-stable.

Figure 2.3 The pink region represents the stability for the second-order Adams-Bashforth method. Reprinted from [27].

An excellent feature of the MATLAB ODE solvers is that they all use the same syntax. The MATLAB syntax of the ODE solvers is described by:

\[
[T, Y] = \text{odesolver}(\text{odefun}, \text{tspan}, y0)
\]

\[
[T, Y] = \text{odesolver}(\text{odefun}, \text{tspan}, yo, \text{options})
\]

\[
[T, Y, TE, YE, IE] = \text{odesolver}(\text{odefun}, \text{tspan}, y0, \text{options})
\]
sol = odesolver (odefun, [t0 tf], y0....)

where the odesolver is one of ode45, ode23, ode113, ode15s, ode23s, ode23t or ode23tb. The following describes the output arguments for the odesolvers:

‘T’ represents the column vector of time points, ‘Y’ represents the solution array and each row in ‘Y’ corresponds to the solution at a time returned in the row corresponding to ‘T’, ‘TE’ denotes the time at which the event occurs, ‘YE’ denotes the solution at the time of the event, ‘IE’ represents the index ‘i’ of the event function that actually vanishes and ‘sol’ represents the structure to evaluate the solution.

2.3.2 Using ODE’s for simulating biomolecular systems

The transient behavior of the concentration of the species in a biomolecular system can be simulated by the ODE model. The steady state of a biological system can be described by a simple first order differential equation. Thus the development of the ordinary differential equation forms a vital part of computational biology. The advantage of simulations is that they help in understanding the overall process and also the effect of parameter variations on the biological system. Since biological experiments are always time consuming and expensive, it is better to simulate the system (biological pathways) using the ODE solvers and thus identify the effect of mutation on the system. As noted above, this process of modeling and simulating biological pathways using computer based techniques is known as an in silico process [6].

In order to describe the mathematical modeling better, let us consider a system with X, Y and Z as three chemical species and k₁ and k₂ as rate constants. This system is represented in the form of two reactions as show in equation (2.19)
Equations (2.20), (2.21) and (2.22) show the representation of the biological system by an ordinary differential equation (ODE) model. These ODE models can be solved by the ODE solvers described in Table 2.1.

However, the ODE models have disadvantages. The ODE models are deterministic models and they do not represent variations in reaction time or reaction mechanism, i.e., they change according to a mathematical formula but not due to some random fluctuations. In order to study the noisy behavior of a system we need stochastic modeling and simulation.

Most of the biological experiments are highly time consuming and also computationally expensive. As a result, to characterize a biological pathway might not always be practical and also necessary. Therefore, a more feasible solution is to simulate the biological system using the ODE solvers and thereby discover the mutation effects on the system. Simulating the system using the ODE solvers can help us identify a smaller subset of reactions which are very significant. Then, the reactions in this smaller subset alone can be modified or mutated, either within the living cell (*in vivo*) or outside the living cell or in the lab (*in vitro*), to verify the simulation results. This process of modeling and simulation of the biological pathways is known as “*in silico*” process [6]. MATLAB also provides us with a built-in toolbox called the System Biology Toolbox which helps in modeling and analyzing the biological systems and pathways.
[28]. The System biology toolbox is a simulation-based mathematical analysis framework for the biological systems. This toolbox allows for the modeling of biological systems and then performs the deterministic or stochastic simulations of the system. One can also import and export Systems Biology Markup Language into the Matlab simulation system using the System Biology Toolbox. The parameter estimation and mathematical parameter sensitivity analysis can be performed using the System Biology Toolbox [29]. Using the System Biology Toolbox, one can automatically generate the ODE’s from the reactions. Then it solves these equations and generates a graphical output. However, this toolbox has its own drawbacks. The System Biology Toolbox depends on certain user-defined rules for parameter estimation and does not include control rules for the biochemical reactions. Another major disadvantage with this toolbox is the cost involved. The Systems biology toolbox is not a free software package that comes along with Matlab but is priced at $3000 [29] for a single user.

2.4 Occurrence of stochasticity in biological systems

Stochastic fluctuations are important in biological systems at all levels [30]. The interactions between molecules like DNA, mRNA and proteins follow the law of physics at the microscopic level of functionality. According to the famous Schrodinger’s $\sqrt{n}$ law [31], the randomness in a system is inversely proportional to the square root of the number of particles, which is an indicator of the system size. Therefore this implies that a lower number of particles results in a high fluctuation. This is very important in biomolecular computing and also in nanoscale computing.

Biochemical species that participate in processes like gene transcription, gene regulation and signal transduction mostly occur in low copy numbers. As a result elementary reactions such as polymerase binding or complex formation take place with widely spread out reaction times.
These stochastic effects that arise due to the inherent nature of the biochemical interactions are quite often termed “intrinsic noise”. There also exists the extrinsic component of noise that arises from random fluctuations in other factors like the number of ribosomes, the stage of the cell cycle, mRNA degradation and the cellular environment. This extrinsic component of noise is mainly due to the external environmental conditions.

Biological systems in general represent a dynamic environment. This implies that the biological systems exhibit changes from one state to another. The exact nature of this dynamics in a biological system is controlled by the order of events modeled in the network. Stochasticity in the dynamics arises in two ways, namely intrinsic stochasticity and extrinsic stochasticity. The intrinsic stochasticity is attached or connected to the system which arises due to the relatively small number of reactant molecules whereas the extrinsic stochasticity arises mainly due to random variations or fluctuations due to one or more environmental conditions like temperature and concentrations of the reactant species [4].

2.5 Stochastic simulation

As already discussed in the previous section, the stochasticity becomes more prominent with the decrease in the number of the molecules. With the increase in concentrations of the reacting species, the stochastic fluctuations become less prominent and the behavior automatically tends towards the deterministic solution. The system dynamics approach of biological systems involves deciphering the time based evolution of several biochemical species in a particular model. There are actually two ways to define the time-evolution of a spatially homogenous chemical system, namely the deterministic approach and the stochastic approach. The deterministic approach considers the time-evolution of the system as a more continuous and predictable phenomenon which is actually governed by a set of coupled, ordinary differential
equations better known as “reaction-rate equations”. The stochastic approach, on the other hand, treats the time-evolution as some sort of random process which is governed by a single differential equation which is known as the “Chemical Master Equation” or “Stochastic Master Equation”. The advantages of the stochastic simulation methods are that they allow one to study the distribution of the species population at a specific time [32].

2.5.1.1 Advantages of stochastic simulation

As already discussed above, a stochastic model is a mathematical model which involves estimating probability distributions of certain potential outcomes by allowing random variations in one or more inputs over a period of time [33]. The major advantages of stochastic modeling or simulation are that it takes the mesoscopic view of the system so that it can keep track of the exact number of molecules in the system that is being modeled. When a system is modeled using stochastic simulation, it keeps track of the small number of molecules of the system under certain reasonable conditions and the system also exhibits oscillating behavior [4].

2.5.1.2 Disadvantages of stochastic simulation

Although the stochastic simulation methods are better when compared to the deterministic approach, these methods have their own set of limitations. Since the stochastic simulation methods keep track of the exact number of molecules in the system, the speed of the stochastic simulation is quite slow when compared to the deterministic approach.

2.5.2 Stochastic models

The changes in molecular species in biological pathways do not occur at regular intervals of time. The deterministic modeling or simulation of biological pathways does not give us accurate results. Therefore, randomness of changes in molecular species and time intervals and
randomness of changes in molecular reactions are included in the biological models leading to more accurate models than deterministic models. In order to analyze chemical reactions that involve a large number of species with complex reaction kinetics, the stochastic simulation algorithms and methods were developed. The first stochastic algorithm that was developed was the Gillespie Algorithm proposed by Dan Gillespie in 1976 [12]. The Gillespie Algorithm is based on the Monte Carlo method. This algorithm mainly generates a correct possible solution of a particular stochastic equation. Gillespie developed and published this algorithm to simulate chemical and biochemical systems of reactions in an efficient and accurate manner while using limited computational power. The Gillespie Algorithm can also be applied to the stochastic modeling of biological systems. With the day to day advancement in the field of computers, this algorithm has been used to simulate increasingly complex systems. Since every reaction is explicitly simulated, the Gillespie Algorithm allows a discrete and stochastic simulation of the system. When this system is simulated, the Gillespie realization of the system represents a random walk which represents the distribution of the master equation. The main idea behind the Gillespie Algorithm is modeling of the collision of the molecules within a reaction vessel [12]. Gibson and Bruck [14] developed the Next Reaction Method which is a modification of the Gillespie Algorithm [10]. The advantage with the Next Reaction Method is its high efficiency and the reduction in simulation time for large biological systems.

2.5.2.1 Gillespie Algorithm

There were two different but equivalent formulations that were developed by Gillespie, the Direct method and the First Reaction Method [12]. In R. Krishnan’s thesis [9] the direct method was used to simulate several biological systems. We will briefly describe the direct method here. The terminology used here is from [34].
a) The first step needed to run the Gillespie Algorithm is the initialization of the number of molecules in the system, the reaction constants and the number of random generators. This is called the **Initialization Step**.

b) The next step involves generating random numbers to identify the next reaction that is to occur and also to identify the time interval. The probability of choosing a given reaction is proportional to the number of substrate molecules. This step is called the **Monte Carlo Step**.

c) Then the time step that was randomly generated in the first step is increased and also the molecule count is updated based on the reaction that had occurred. This step is called the **Update Step**.

d) If the number of reactants is not zero and if the simulation time has not been exceeded, then repeat from step b. This step is called the **Iteration Step**.

Since the Gillespie Algorithm is computationally expensive, many more efficient simulation techniques based on its ideas have been proposed. The various simulation techniques that have been proposed are the Next Reaction Method (by Gibson and Bruck), Tau-Leaping method, the Optimized Direct Method (ODM), the Sorting Direct Method (SDM) and the Logarithmic Direct Method as well as several hybrid techniques where abundant reactants are modeled by deterministic behavior [10], [15].

### 2.5.2.2 Next Reaction Method

As described in the previous section, Gillespie developed two different but equivalent formulations: the Direct method and the First Reaction Method. Gibson and Bruck’s Next Reaction Method is an adaptation of the first reaction method. The Next Reaction Method is comparatively faster than the First Reaction Method and it is also considered to be much more
efficient than the direct method when the system involves many species and loosely coupled reaction channels [15]. In the Next Reaction Method, instead of computing the time to each reaction, one deals with the time at which a reaction takes place. These times are not always computed each and every time but maybe re-used by storing them in an indexed priority queue. Updates are determined by a dependency graph which shows the dependencies among the reactions under consideration. The summary of the algorithm for the Next Reaction Method is given below:

1) Initialization:
   a) The number of molecules for all the species is initialized, the initial time is set to zero and a dependency graph is generated.
   b) The propensity functions \((a_i)\) are computed for all the reactions.
   c) The reaction times for all the reactions are computed using the formula:

   \[ T = \frac{1}{a_i} \ln \left( \frac{1}{r} \right). \]

   Where, ‘r’ is a random number.
   d) The values of the time and the propensity functions are stored.

2) The next reaction that occurs is determined and the reaction is carried out. Also, the time is updated.

3) a) The propensity functions for the reactions whose substrate numbers have changed or those specified by the dependency graph are updated.
   b) The times for those reactions whose propensity functions were updated are also updated.

4) The iterations are repeated from step 2 until the simulation completes.
2.6 Other stochastic simulation methods

In the previous sections, we discussed the Gillespie Algorithm in detail and also introduced various other stochastic simulation techniques. Each of these simulation techniques imposes a certain constraint on the computational power, the working knowledge of the system and also the input of the numerical parameters involved in the system. Further, these simulation techniques provide different generalizations of the system and each simulation technique produces solutions of varying accuracy. The following table gives an overview of the various simulation techniques.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Computational Cost</th>
<th>Modeling Knowledge</th>
<th>Speed</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gillespie Algorithm [10, 1977]</td>
<td>Very High</td>
<td>Medium</td>
<td>Slow</td>
<td>Very high</td>
</tr>
<tr>
<td>Gibson and Bruck Algorithm [14, 2000]</td>
<td>High</td>
<td>Medium</td>
<td>Fast</td>
<td>Very High</td>
</tr>
<tr>
<td>Tau-Leap Methods [36, 2001]</td>
<td>Low</td>
<td>High</td>
<td>Very Fast</td>
<td>Medium</td>
</tr>
<tr>
<td>Parallel Algorithms [38, 1996]</td>
<td>Low</td>
<td>Medium</td>
<td>Very fast</td>
<td>Low</td>
</tr>
</tbody>
</table>

Table 2.2 Comparison of different stochastic algorithms [4]
Table 2.2 briefly describes the comparisons between various stochastic algorithms. From the table we observe that both the Spatial-temporal Algorithms and the Gillespie Algorithms have the highest computational cost and as a result they are equally slow in simulation. However, the accuracy of these simulation methods is very high. The Tau-leap methods, parallel algorithms and hybrid algorithms have high speed. However, these simulation methods require very high modeling knowledge. The Stochsim and Gibson and Bruck algorithms are the most efficient algorithms when compared to all the simulation techniques and these algorithms have very high speed of simulation without compromising the accuracy of the solutions [4]. In this thesis we mainly discuss the application of the Gibson and Bruck algorithm to the two biological systems Phage Lambda and Bioluminescence.

Apart from the different stochastic algorithms mentioned in Table 2.2, several environments and software tools for the stochastic simulation of biological systems have been developed. Some of the software tools that perform the stochastic simulation of biological systems include Cain [42], Synbioss [43], SBML [44], Dizzy [45], FERN [46], stochastic simulation using MATLAB [47], and STOCKS [48]. Some of these software tools have been designed using complex object oriented programming. After careful investigation and study we have decided to perform the stochastic simulation of the biological pathways by writing a code in Python using Gibson and Bruck’s Next Reaction Method. The template created in Python using the Next Reaction Method is easy to use and modify for any biological system. This algorithm also provides us with higher accuracy and faster simulation time.

2.7 Agent Based Modeling (ABM)

In this section we will briefly discuss another simulation paradigm, Agent Based Modeling, and its advantages and disadvantages. Most of the biological systems that we are interested in
simulating are quite complex in nature. These complex biological systems have several components that are coupled in a non-linear fashion. The variables in these complex systems exhibit discontinuous and complicated behavior over time. Many of these complex systems exhibit the emergence property. This emergence property describes the formation of complex patterns from simple interaction rules.

An agent-based model (ABM) is a computer based model used for simulating dynamical systems of certain autonomous decision making entities called agents [49]. The main idea is to construct the model using these autonomous decision making entities and to simulate the interactions of these agents in parallel to model the real phenomena on a system level. This process is one of the ways that complex systems and patterns arise out of a multiplicity of relatively simple interactions from the micro level of the system to the macro level of the system. Thus ABM is a relatively new computational modeling paradigm. Many modern software techniques are implemented for developing software for the agent based model, including decomposition or dividing the problem into several small, manageable parts, generalization by reducing the information content of a particular concept, choosing which particular details of the problem to model and identifying and managing the relationships among the various system components [50], [51]. Thus, the ABM is a bottom-up approach where the lowest entities as described above are called the agents which interact among themselves autonomously. ABM has been used for biological systems, for example for the Phage Lambda system in [52]. We will compare the results from V. Vallurupalli’s thesis [13] with the results we obtain using the Next Reaction Method.
2.7.1 Advantages of Agent Based Modeling (ABM)

To model a biological system such as Phage Lambda using the differential equations approach as described in [6, 53], we require that all the different processes be converted into chemical equations which are later converted into differential equations. One of the major limitations or drawbacks with these differential equations is that they do not take into account the spatial aspects of the system involved. Partial differential equations can be employed but they add even more complexity. However ABM takes into account these spatial and temporal aspects and as a result can be considered to be closer to the actual biological processes [13]. The major advantages of ABM over other modeling techniques is that it can capture an emergent phenomenon, it provides what is called the natural description of the system involved and above all ABM is very flexible [49]. With an agent based approach, many processes work simultaneously and as a result complex and unpredictable behavior can often be observed. However, ABM also has some drawbacks.

2.7.2 Drawbacks of ABM

The ABM is actually a bottom-up approach and to model a system with the bottom-up approach requires that every individual agent’s behavior be described in detail. The more details we require about the behavior of each individual agent, the greater the computational power that is required for simulating the behavior of all these agents. However this is a limitation only while modeling large systems using ABM [49].

2.8 Limitations of biological model simulation

In this thesis, we discuss the mathematical modeling and the stochastic modeling of the biological systems or pathways. The mathematical modeling of biological pathways has two
limitations. The first limitation is the availability of the data that is required for simulation. The Ordinary Differential Equation (ODE) models require certain parameter data such as the species concentration and the rate constants. The concentration of species can be easily procured but obtaining the rate constants remains the most constraining step as they are not readily available for most biological systems [21]. The rate constants are empirical data and these are mostly obtained through experimentation.

The other limitation is to establish a link between the biological experiments and the mathematical modeling and simulation. Most of the current research being carried out focuses on the conversion of the biological systems into mathematical models and the software currently available in the market also works on them. Though mathematical modeling is essential in understanding the biological function of the system, the conversion of mathematical modeling and simulation results into biological controls has not been addressed so far [9].
3 Modeling the Phage Lambda System

3.1 Introduction to biological pathways

As already discussed in Chapter 1, biological pathways are networks of certain biological processes. In other words, biological pathways model actions among the molecules in a cell that actually lead to a certain product or sometimes even a change in the cell. These biological pathways model how biological molecules interact in order to achieve a biological function and also how they respond to external stimuli. The pathways are derived through scientific experimentation and data analysis and these pathways also capture the current knowledge of the biological process. In most cases a pathway is defined as a network of biological interactions. Generally pathways are represented as graphs, consisting of nodes and edges as in Figure 3.1.

Some biological pathways which are of interest in computing are the Phage Lambda system and the Bioluminescence system. The Phage Lambda system is a system which exhibits two states—Lysis and Lysogeny and these are governed by strong interlocking feedback loops and several
other regulatory mechanisms such as cell replication, transcription and translation. In order to better describe these regulatory mechanisms, the Phage Lambda system is analyzed and modeled computationally. In this chapter we describe the Phage Lambda system, the deterministic modeling of the Phage Lambda system and the stochastic modeling and simulation of the Phage Lambda system. The deterministic modeling is carried out using Matlab and the stochastic simulation is implemented using the Next Reaction Method by forming templates based on Python programming. The templates are formed in such a way that they can be extended to any biological system. In chapter 4 we describe another useful system, Bioluminescence.

### 3.2 Phage Lambda system

Phage Lambda is a virus that infects *Escherichia coli* (*E.coli*) bacteria [55] [56]. The *E.coli* bacterium is a Gram negative bacterium that does not retain the crystal violet dye in the gram staining protocol and is commonly found in the intestine of certain warm-blooded organisms [57]. When an *E.coli* bacterium which has been infected with a Phage Lambda virus, is exposed to a dose of ultraviolet light, it produces a burst of the viruses. These viruses then infect fresh bacteria. This process by which the Phage Lambda viruses reproduce rapidly is called **Lysis**. However, in the absence of the UV light, the infected bacterium grows and divides naturally. In other words, the Phage Lambda virus remains in its inactive form. This process by which the Phage Lambda reproduces passively is called **Lysogeny** [9, 13]. Therefore, the Phage Lambda exhibits only two stages, the Lytic state and the Lysogenic state. These are related to the presence or absence of two specific proteins, $R_p$ and Cro.
The Phage Lambda switches between the Lytic state and the Lysogenic state and this switching mechanism is quite similar to the ON and OFF states of a digital inverter. According to Jacob and Monod [58], this switching mechanism is a basic example for turning “on” and “off” of genes. The discovery of this switching mechanism has contributed to a new field of study namely bio-circuit design [53, 59].

With the switching mechanism of the Phage Lambda, an analogy can be derived between it and the electrical switches. Considering an inverter circuit, when the input signal is high, the output is low, and when the input signal is low, the output is high. Similarly in a Phage Lambda system, when a repressor protein $R_p$ (input) is present, there isn’t any Lytic growth and hence the protein Cro (output) is not synthesized. However, in the absence of $R_p$, Lytic growth occurs and as a result the protein Cro (output) is synthesized. This forms the basis for the operation of a bio-inverter. This analogy is shown in Figure 3.3.

**Figure 3.2** Lysis plaques of Lambda Phage in *E.coli* bacteria [57].
Now, we discuss in brief how the gene expression as it can be clearly visualized in the switching between the Lytic state and the Lysogenic state. Expression of the genes is an important factor in deciding if a gene is on or off. The genes which are actually expressed are considered to be on whereas those genes which are not expressed are said to be off [45]. Just as electrical circuits are modeled using the order differential equations; bio-inverter can also be modeled in a similar way except for the fact that the bio-inverter is a living cell. The different reactions that occur in a living cell are dimerization reactions, transcription of the DNA to mRNA, translation of the mRNA to proteins and the decay reactions. Therefore, the bio-inverter can be modeled using rate equations, chemical constants and kinetic constants.

### 3.3 Modeling the Phage Lambda system

Modeling of the Phage Lambda system is based on the work done by Weiss [53, 59] and Krishnan [6, 9]. The basic input to the Phage Lambda system is the repressor protein or Rp which forms the dimers. These dimers then bind with the appropriate gene, thereby preventing
the formation of the output protein or Cro. There are induction processes like the UV light and Isopropyl β- D- 1- thiogalactopyranoside (IPTG) which result in the breakage of the repressor dimers and also enable the transcription and the translation of the output protein (Cro) [9]. This entire system has been modeled using ten different reactions and eight different biochemical species as described below.

The first reaction describes the dimerization of the input to the Phage Lambda system or the input repressor protein,

\[
R_p + R_p \rightarrow R_{p2} \quad (3.1)
\]

The second reaction describes the breakdown of the input repressor protein,

\[
R_{p2} \rightarrow R_p + R_p \quad (3.2)
\]

The third reaction describes the decaying process or the decay reaction of the repressor monomer,

\[
R_p \rightarrow \text{decay} \quad (3.3)
\]

The fourth reaction describes the decay reaction of the repressor dimer,

\[
R_{p2} \rightarrow \text{decay} \quad (3.4)
\]

The fifth reaction represents the repressor reaction,

\[
R_{p2} + C_g \rightarrow C_{gR_{p2}} \quad (3.5)
\]

The sixth reaction indicates the disassociation reaction,

\[
C_{gR_{p2}} \rightarrow C_g + R_{p2} \quad (3.6)
\]
The seventh reaction represents the transcription reaction, from DNA to RNA,

\[ k_7 \quad C_g + RNA_{poly} \rightarrow C_g + RNA_{poly} + mRNA \]  

(3.7)

The eighth reaction represents the translation reaction, from RNA to protein,

\[ k_8 \quad mRNA + rRNA \rightarrow mRNA + rRNA + Cro \]  

(3.8)

The ninth reaction indicates the decaying of the mRNA,

\[ k_9 \quad mRNA \rightarrow \text{decay} \]  

(3.9)

The output protein decay is represented by the tenth reaction,

\[ k_{10} \quad Cro \rightarrow \text{decay} \]  

(3.10)

The above reactions are biochemical reactions that model a bio-inverter.

### 3.4 Differential equations modeling the Phage Lambda system

According to the reactions described in the previous section, the bio-inverter model includes eight species, $R_p$, $R_{p2}$, $C_g$, $C_gR_{p2}$, mRNA, pRNA, rRNA and Cro. From section 2.3.2, based on the equations (2.19) to (2.22), we can write the ODE models of all the ten biochemical reactions mentioned in the previous section.

**Reaction 1:**

\[ k_1 \quad R_p + R_p \rightarrow R_{p2} \]  

(3.11)

**Reaction 2:**

\[ k_2 \quad R_{p2} \rightarrow R_p + R_p \]  

(3.12)

The ODE template for the above reactions can be obtained by considering it to be a one input-one output reversible system. The reactions 1 and 2 are similar to the equation (3.13) described in the next page:
\[ \text{A} + \text{A} \leftrightarrow \text{C} \] (3.13)

The ODE template for equation (3.13) is described below:

\[
\frac{d[C]}{dt} = k_1.[A]^2 - k_2.[C] 
\] (3.14)

\[
\frac{d[A]}{dt} = -2.k_1.[A]^2 + 2.k_2.[C] 
\] (3.15)

Based on the equations (3.14), (3.15), the ODE models for the reactions 1 and 2 can be obtained as:

\[
\frac{d[R_{p2}]}{dt} = k_1.[R_p]^2 - k_2.[R_{p2}] 
\] (3.16)

\[
\frac{d[R_p]}{dt} = -2.k_1.[R_p]^2 + 2.k_2.[R_{p2}] 
\] (3.17)

**Reaction 3:**

\[
R_p \rightarrow R_{p\text{decay}} 
\] (3.18)

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 3 is similar to the equation (3.19) as described in the next page.

\[
\text{A} \rightarrow \text{decay} 
\] (3.19)

The ODE template for the equation (3.19) is given as:

\[
\frac{d[A]}{dt} = -k_1.[A] 
\] (3.20)

Based on equation (3.20), the ode model for the reaction 3 can be described as:

\[
\frac{d[R_p]}{dt} = -k_3.[R_p] 
\] (3.21)
Reaction 4:
\[ k_4 \]
\[ R_{p2} \rightarrow R_{p2\text{decay}} \]  \hspace{1cm} (3.22)

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 4 is similar to the equation (3.19).

The ODE model for the reaction 4 is similar to the ODE template described in equation (3.20) and is described below:

\[ \frac{d[R_{p2}]}{dt} = -k_4[R_{p2}] \]  \hspace{1cm} (3.23)

Reaction 5:
\[ k_5 \]
\[ R_{p2} + C_g \rightarrow CgR_{p2} \]  \hspace{1cm} (3.24)

Reaction 6:
\[ k_6 \]
\[ C_gR_{p2} \rightarrow C_g + R_{p2} \]  \hspace{1cm} (3.25)

The ODE template for the above reactions can be obtained by considering it to be two input-one output reversible systems. The reactions 5 and reactions 6 are similar to the equation (3.26) described below.

\[ k_1 \]
\[ A + B \rightarrow C \]  \hspace{1cm} (3.26)

\[ k_2 \]

The ODE template for equation (3.26) is described below:

\[ \frac{d[A]}{dt} = -k_1[A][B] + k_2[C] \]  \hspace{1cm} (3.27)

\[ \frac{d[B]}{dt} = -k_1[A][B] + k_2[C] \]  \hspace{1cm} (3.28)

\[ \frac{d[C]}{dt} = k_1[A][B] - k_2[C] \]  \hspace{1cm} (3.29)

Based on the equations (3.27) to (3.29), the ODE models for the reactions 5 and 6 can be obtained as:
\[ \frac{d[R_{p2}]}{dt} = -k_5[R_{p2}][C_g] + k_6[R_{p2}][C_g] \]  
(3.30)

\[ \frac{d[C_g]}{dt} = -k_5[R_{p2}][C_g] + k_6[R_{p2}][C_g] \]  
(3.31)

\[ \frac{d[C_g R_{p2}]}{dt} = k_5[R_{p2}][C_g] - k_6[R_{p2}][C_g] \]  
(3.32)

**Reaction 7:**

\[ k_7 \]

\[ C_g + RNA_{poly} \rightarrow C_g + RNA_{poly} + mRNA \]  
(3.34)

The ODE template for the above reactions can be obtained by considering it to be two input-one output reversible systems. The ODE template is similar to what was described in equations (3.27) to (3.29). Based on those equations, ODE model for the reaction 7 is described as:

\[ \frac{d[mRNA]}{dt} = k_7[C_g][RNA_{poly}] \]  
(3.35)

**Reaction 8:**

\[ k_8 \]

\[ mRNA + rRNA \rightarrow mRNA + rRNA + Cro \]  
(3.36)

The ODE template for the above reactions can be obtained by considering it to be two input-one output reversible systems. The ODE template is similar to what was described in equations (3.27) to (3.29). Based on those equations, ODE model for the reaction 8 is described as:

\[ \frac{d[Cro]}{dt} = k_8[mRNA][rRNA] \]  
(3.37)

**Reaction 9:**

\[ k_9 \]

\[ mRNA \rightarrow mRNA_{decay} \]  
(3.38)

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 9 is similar to the equation (3.19).
The ODE model for the reaction 9 is similar to the ode template described in equation (3.20).

\[ \frac{d[mRNA]}{dt} = -k_9[mRNA] \quad (3.39) \]

**Reaction 10:**

\[ k_{10} \]

\[ \text{Cro} \rightarrow \text{Cro}_{\text{decay}} \quad (3.40) \]

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 10 is similar to the equation (3.19).

The ODE model for the reaction 10 is similar to the ODE template described in equation (3.20).

\[ \frac{d[\text{Cro}]}{dt} = -k_{10}[\text{Cro}] \quad (3.41) \]

The ODE models that are generated for individual reactions are then combined to generate the ODE model for the entire Phage Lambda System.

The ODE model of the entire Phage Lambda System is given as follows:

\[ \frac{d[R_p]}{dt} = -2. k_1[R_p]^2 + 2. k_2[R_{p2}] - k_3[R_p] \quad (3.42) \]

The above equation is obtained by combining the equations (3.17) and (3.21).

Now, combining the equations (3.16), (3.23) and (3.30), we get;

\[ \frac{d[R_{p2}]}{dt} = k_{11}[R_p]^2 - k_2[R_{p2}] - k_{41}[R_{p2}] - k_5[R_{p2}][C_g] + k_6[C_g R_{p2}] \quad (3.43) \]

From the equation (3.31), we get the following equation

\[ \frac{d[C_g]}{dt} = -k_5[R_{p2}][C_g] + k_6[C_g R_{p2}] \quad (3.44) \]
From the equation (3.32), we get the following equation

\[
d[C_g \text{ R}_p]/dt = k_5[R_p2][C_g] - k_6[C_g \text{ R}_p]
\]  \hspace{2cm} (3.45)

From the equations (3.35) and (3.39), we get the following equation

\[
d[mRNA]/dt = k_7[C_g][RNA_{poly}] - k_9[mRNA]
\]  \hspace{2cm} (3.46)

From the equations (3.37) and (3.41), we get the following equation

\[
d[Cro]/dt = k_8[mRNA][rRNA] - k_{10}[Cro]
\]  \hspace{2cm} (3.47)

Equations (3.42) to (3.47) give the ODE model of the entire Phage Lambda system [9].

### 3.5 Mathematical simulation of the Phage Lambda system

From the previous two sections, we know that the bio-inverter model of the Phage Lambda system consists of ten biochemical reactions and eight different species. These species are Rp, R_p2, mRNA, pRNA, rRNA, Cro, Cg and CgR_p2. While carrying the mathematical simulation for the Phage Lambda system, we assume that the concentrations of pRNA, rRNA and Cg are constant. The values of pRNA and rRNA are taken to be 1 Molar and the concentration of Cg is taken as 0.07µM. The Phage Lambda system consists of ten reactions and also involves ten rate constants as described in the previous sections. The values of the rate constants \(k_1\) to \(k_{10}\) are obtained from Weiss and Ptashne [59, 60]. The rate constants that are used for modeling the Phage Lambda system are given in the Table 3.1. In order to maintain a constant supply of the input protein (Rp), we add a term called the “drive” term to the rate equation that involves the input protein (Rp), as suggested by Weiss [53].
The relation between the input protein to the Phage Lambda System, repressor \( (R_p) \) and the output protein \( (Cro) \) is expressed by the following equation,

\[
Cro = \frac{(k_7.k_8. [pRNA]. [rRNA].C_g)}{(k_9.k_{10}. (1 + (k_5 / k_6).R_p))}
\] (3.48)

The Phage Lambda system is modeled mathematically by using the ODE’s. R. Krishnan [9], had used the ode15s solver to solve the ODE’s developed for the biochemical reactions of the Phage Lambda system. However, we use the Matlab ODE solver ode45 and ode113 for solving the ordinary differential equations that have been developed for the Phage Lambda system. The ODE’s solve the differential equations at certain fixed intervals of time as determined by the time step specified. The ode45 and ode113 though used for non-stiff equations are very accurate when compared to the ode15s solver. Stiffness is actually an efficiency issue and if we are not concerned with the computation time, we then would not be concerned with stiffness either.

### 3.6 Stochastic method implementation for the Phage Lambda system

The major disadvantage with the ODE model is that it does not consider any fluctuations in the biological system and it assumes that the process is deterministic. Generally in any biological pathway, the changes that take place in the molecular species are not deterministic in nature and therefore incorporating some amount of noise or randomness to the pathway might play an

<table>
<thead>
<tr>
<th>Rate</th>
<th>( K_1 )</th>
<th>( K_2 )</th>
<th>( K_3 )</th>
<th>( K_4 )</th>
<th>( K_5 )</th>
<th>( K_6 )</th>
<th>( K_7 )</th>
<th>( K_8 )</th>
<th>( K_{10} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>8.333</td>
<td>0.1667</td>
<td>0.5775</td>
<td>0.5775</td>
<td>66.67</td>
<td>0.2</td>
<td>0.0001</td>
<td>0.03</td>
<td>2.0</td>
</tr>
</tbody>
</table>

Table 3.1 Rate constants for the Phage Lambda system
important role in deriving accurate results for the biological system. R. Krishnan [9] has developed a stochastic model to simulate the Phage Lambda system using the Gillespie Algorithm. In our thesis, we have developed a stochastic model to simulate the Phage Lambda system using Gibson and Bruck’s Next Reaction Method. Gibson and Bruck’s Next Reaction Method is an improvement over the Gillespie’s First Reaction Method. As explained in the earlier sections, the Phage Lambda system exhibits characteristics similar to that of an electrical inverter. The simulation of the stochastic model is carried out to verify the inverter characteristics and it may also be used to test the robustness of the bio-inverter in the presence of noise and also due to the random order of the biochemical reactions. McAdams and Arkin [61] have explained the behavior of the Phage Lambda lysis-lysogeny pathway. Their description included the models of the operator and promoter sites and they also produced a complex and complete model of the lysis-lysogeny pathway and the reactions in it [9]. Modeling and simulation of a stochastic process like the lambda lysis-lysogeny pathway is quite complex and also difficult. Hence, we are more interested in checking and analyzing the simplified model of a bio-inverter from a circuit perspective. The main idea behind developing a stochastic model is to test the robustness of the bio-inverter in the presence of noise which gives us scope to test the transient characteristics of the bio-inverter in the presence of noise. In chapter 4, the algorithm that was used by R. Krishnan [9] in his thesis for modeling the Bioluminescence system and also the algorithm that is being used in our thesis for modeling the same system are described.
4 Modeling the Bioluminescence System

4.1 Bioluminescence system

Bioluminescence refers to the production and emission of light by living organisms. This light is produced by the chemical reactions which convert the chemical energy to light. Bioluminescence typically occurs in the marine vertebrates and invertebrates and also in certain microorganisms and terrestrial animals. Most of the deep-sea marine animals produce bioluminescence in one or the other form. One such organism that produces bioluminescence is the *Vibrio Fischeri* (*V.Fischeri*). The *V.Fischeri* is a gram-negative bacterium that is commonly found in the marine environment and is a rod-shaped bacterium. Gram negative bacteria are those bacteria which do not retain the crystal violet dye during an empirical method of differentiating the bacteria into two large groups which is called the gram staining protocol [9]. Figure 4.1 shows the structure of the cell wall of a gram negative bacterium.

![Figure 4.1 Structure of a gram negative cell wall. Reprinted from [62]](image)
The bioluminescence in bacteria is related to a protein complex called Lux. The bacteria that exhibit bioluminescence can communicate and hence the bioluminescence system is useful in communication. The luminescence in the *V. Fischeri* bacterium is obtained by a process called the quorum sensing. Quorum sensing is the process by which the gene expression is regulated in response to the fluctuations that take place in the cell-population density [63]. The quorum sensing was first observed in the *V. Fischeri* bacterium. The basic model of a *V. Fischeri* cell is shown in the Figure 4.2.

![Figure 4.2 Basic model of the *V. Fischeri* system. Reprinted from [9].](image)

As seen in Figure 4.2, the auto-inducer forms the input for a single *V. Fischeri* cell and the biochemical reactions that take place inside the cell release both the auto-inducer and the Lux A/B as the output. The luminescence exhibited by the *V. Fischeri* cell is produced by the Lux A/B protein.

Several studies to find the biochemical reactions that describe the bioluminescence system have led to the identification of two main models: the hybrid model that was developed by Dunlap et al. [64] and the other model was the biochemical reaction model that was developed by Cox et al. [65]. These two models are described in the subsequent sections.
4.1.1 Application of the Bioluminescence system

The Bioluminescence system is useful from both the scientific and the engineering point of view. For a signal to be recognized, the auto-inducer must pass a threshold value. Bioluminescence is very important because the system exhibiting bioluminescence sends out a constant signal in addition to a controlled signal. For engineering applications, it is quite useful because there are several pathways which are similar. Weiss and Knight’s [66] work presented an effective signaling model and this signaling model mainly functions as a basic computing device.

Ever since the research of the biological pathway of producing light in bacteria began two decades ago, interest and research on applications in bioengineering for luminescence has increased significantly [66]. The Bioluminescence system is useful for communication systems and technology. The inducers and the co-repressors are termed effectors and these effectors are used mainly because they provide faster intracellular interactions and intercellular communications.

4.1.2 Model- I of the Bioluminescence system

This model is better known as the hybrid mathematical model and was developed by Dunlap et al [64]. The hybrid model is obtained by approximating several functions. In order to model the luminescence of the V. Fischeri cell, the hybrid model restricts itself to around nine parameters as described below:

\[x_0 = \text{scaled population},\]
\[x_1 = \text{mRNA of LuxR},\]
\[x_2 = \text{mRNA of LuxICDABEG},\]
\[x_3 = \text{LuxR},\]
\( x_4 = \text{LuxI}, \)
\( x_5 = \text{LuxA/B}, \)
\( x_6 = \text{LuxC/D/E}, \)
\( x_7 = \text{auto-inducer Ai}, \)
\( x_8 = \text{LuxR-Ai complex}. \)

The bioluminescence process in a \textit{V. Fischeri} cell occurs mainly due to the production and release of certain chemical molecules called the auto-inducers. The concentration of these light weight chemical molecules increases as a function of the increasing cell population density [5]. These auto-inducer molecules are produced by the LuxI protein inside the bacterial cells. The auto-inducer molecules that play a vital role in producing the bioluminescence phenomenon in the \textit{V. Fischeri} cell are called N-acyl homoserine lactones (AHL). These molecules act as input to the bacterial cells and diffuse through each of these bacterial cells to bind with the LuxR protein and form complex molecules. These complex molecules bind to the Lux regulatory genes and regulate and activate the transcription of LuxICDABEG. The binding of this complex also plays an important role in the negative regulation of the transcription of LuxR protein. There also exists another protein CRP which positively regulates the transcription of LuxR and negatively regulates the transcription of LuxICDABEG. The system assumes a constant concentration for the CRP protein and hence the same is used in the mathematical model [5] [9].
Figure 4.3 Model-I depicting the Bioluminescence process. Reprinted from [9].

4.1.3 Model-II of the Bioluminescence system

The model-II of the V. Fischeri cell describing the Bioluminescence process was developed by Cox et al [65]. The model has 23 biochemical reactions and 23 rate constants. This model includes more details than the hybrid model described in the previous section. The Bioluminescence process in this model is described as follows. The chemical molecule called the auto-inducer binds itself to the LuxR protein to form the LuxR-Ai complex which is denoted by the symbol [LrAi]. This complex forms dimers with itself to obtain [LrAi₂]. The dimer then binds with LuxD to form the LuxD_{complex}. The LuxD_{complex} binds with the OL operon, thereby negatively regulating the formation of LuxR protein. The LrAi₂ complex obtained binds itself to the Lux gene, thereby positively regulating the formation of LuxI protein [9] [65].

This reaction model developed by Cox et al. includes the transcription and translation reactions of the proteins LuxR and LuxI and also the associated decay reactions. The V. Fischeri cell produces the auto-inducer chemical molecule from the LuxI protein. Just like the hybrid model, this model also assumes the initial condition where the CRP protein is bound to the CRP binding site. This binding positively regulates the transcription of the LuxR protein.
4.1.4 Model used in this research

As observed from the previous two sections, model-II has more details than model-I, such as the binding of the LuxD protein to the LrA$i_2$ complex. Also, model-I is obtained by approximating functions and does not list any biochemical reactions. However, model-II lists all the biochemical reactions. Therefore, model-II is used for our simulations with the addition of two more reactions. It is assumed that the production of the luciferase subunit LuxA/B is the same as the production of the LuxI protein. The decay reaction for the luciferase reaction is also introduced. Thus, the Bioluminescence model in our research includes 24 biochemical reactions and each of these reactions is controlled by the respective rate constants [9]. In this chapter we describe the Bioluminescence system, the deterministic modeling of the Bioluminescence system and the stochastic modeling and simulation of the Bioluminescence system. The deterministic modeling is carried out using Matlab and the stochastic simulation is implemented using the Next Reaction Method by forming templates based on Python programming. This template is formed in such a way that it can be extended to any biological system.
4.2 Differential equations modeling the V. Fischeri system

The V. Fischeri system uses the model used by R. Krishnan [9]. In this section we describe the biochemical reactions that are involved in this system and also the ODE equations obtained by modeling the V. Fischeri system. The ODE models of all the biochemical reactions are written based on the equations (2.19) to (2.22) from section 2.3.2. The biochemical reactions and their corresponding ODE models are described below:

**Reaction 1:**

\[ k_1 \]
\[ A_i + \text{LuxR} \rightarrow \text{LrA}_i \]  \hspace{0.5cm} (4.1)

**Reaction 2:**

\[ k_2 \]
\[ \text{LrA}_i \rightarrow A_i + \text{LuxR} \]  \hspace{0.5cm} (4.2)

The ODE templates for the above reactions are obtained by considering the reactions to be a reversible two input-one output system. Reaction 1 and reaction 2 are similar to the equation (4.3) described below:

\[ k_1 \]
\[ A + B \rightarrow C \]  \hspace{0.5cm} (4.3)

\[ k_2 \]

The ODE template for equations (4.3) and (4.4) are described below:

\[ \frac{d[A]}{dt} = -k_1[A][B] + k_2[C] \]  \hspace{0.5cm} (4.4)

\[ \frac{d[B]}{dt} = -k_1[A][B] + k_2[C] \]  \hspace{0.5cm} (4.5)

\[ \frac{d[C]}{dt} = k_1[A][B] - k_2[C] \]  \hspace{0.5cm} (4.6)

Based on the equations (4.4), (4.5) and (4.6) the ODE models for the reactions 1 and 2 can be obtained as:

\[ \frac{d[\text{LrA}_i]}{dt} = -k_2[\text{LrA}_i] + k_1[A_i][\text{LuxR}] \]  \hspace{0.5cm} (4.7)
\[ \frac{d[A_i]}{dt} = -k_1[A_i][LuxR] + k_2[LrA_i] \quad (4.8) \]

\[ \frac{d[LuxR]}{dt} = -k_1[A_i][LuxR] + k_2[LrA_i] \quad (4.9) \]

**Reaction 3:**

\[
k_3 \quad \text{LrA}_i + \text{LrA}_i \rightarrow \text{LrA}_{i2} \quad (4.10)
\]

**Reaction 4:**

\[
k_4 \quad \text{LrA}_{i2} \rightarrow \text{LrA}_i + \text{LrA}_i \quad (4.11)
\]

The ODE templates for the above reactions are obtained by considering the reactions to be a reversible one input-one output system. The reactions 3 and 4 are similar to the equation (4.12) described below:

\[
k_1 \quad A + A \rightarrow C \quad \text{ } (4.12)
\]

\[
k_2
\]

The ODE template for the equation (4.12) is described below:

\[ \frac{d[C]}{dt} = k_1[A]^2 - k_2[C] \quad (4.13) \]

\[ \frac{d[A]}{dt} = -2. k_1[A]^2 + 2. k_2[C] \quad (4.14) \]

Based on the equations (4.13), (4.14), the ODE models for the reactions 3 and 4 can be obtained as:

\[ \frac{d[LrA_{i2}]}{dt} = k_3[LrA_i]^2 - k_4[LrA_{i2}] \quad (4.15) \]

\[ \frac{d[LrA_i]}{dt} = -2. k_3[LrA_i]^2 + 2. k_4[LrA_{i2}] \quad (4.16) \]
Reaction 5: \[ k_5 \]
\[
\text{LrA}_{12} + \text{LuxD} \rightarrow \text{LuxD}_{\text{complex}}
\] (4.17)

Reaction 6: \[ k_6 \]
\[
\text{LuxD}_{\text{complex}} \rightarrow \text{LrA}_{12} + \text{LuxD}
\] (4.18)

The ODE templates for the above reactions are obtained by considering the reactions to be a reversible two input-one output system. The reactions 5 and 6 are similar to the equation (4.3).

Based on the equations (4.4), (4.5) and (4.6), the ODE models for the reactions 5 and 6 can be obtained as:

\[
d[\text{LuxD}_{\text{complex}}]/dt = k_5.[\text{LrA}_{12}][\text{LuxD}] - k_6.[\text{LuxD}_{\text{complex}}]
\] (4.19)

\[
d[\text{LrA}_{12}]/dt = -k_5.[\text{LrA}_{12}][\text{LuxD}] + k_6.[\text{LuxD}_{\text{complex}}]
\] (4.20)

\[
d[\text{LuxD}]/dt = -k_5.[\text{LrA}_{12}][\text{LuxD}] + k_6.[\text{LuxD}_{\text{complex}}]
\] (4.21)

Reaction 7: \[ k_7 \]
\[
\text{LuxD}_{\text{complex}} + \text{ipromR} \rightarrow \text{DNA}_{\text{loop}}
\] (4.22)

Reaction 8: \[ k_8 \]
\[
\text{DNA}_{\text{loop}} \rightarrow \text{LuxD}_{\text{complex}} + \text{ipromR}
\] (4.23)

The ODE templates for the above reactions are obtained by considering the reactions to be a reversible two input-one output system and the ODE model for the above reactions are obtained based on the equations (4.4) to (4.6).

\[
d[\text{DNA}_{\text{loop}}]/dt = k_7.[\text{LuxD}_{\text{complex}}][\text{ipromR}] - k_8.[\text{ipromR}]
\] (4.24)

\[
d[\text{LuxD}_{\text{complex}}]/dt = -k_7.[\text{LuxD}_{\text{complex}}][\text{ipromR}] + k_8.[\text{DNA}_{\text{loop}}]
\] (4.25)

\[
d[\text{ipromR}]/dt = -k_7.[\text{LuxD}_{\text{complex}}][\text{ipromR}] + k_8.[\text{DNA}_{\text{loop}}]
\] (4.26)
Reaction 9: 
\[ k_9 \]
Luxbox + LrA\textsubscript{i2} + promI + promR → ipromI + ipromR

(4.27)

Reaction 10: 
\[ k_{10} \]
ipromI + ipromR → luxbox + LrA\textsubscript{i2} + promI + promR

(4.28)

The ODE templates for the above reactions are obtained by considering the reactions to be a reversible four input-two output system. The reactions 9 and 10 are similar to the equation (4.29) described below:

\[ k_1 \]
\[ A + B + C + D \rightarrow E + F \]

(4.29)

\[ k_2 \]

The ODE template for the equation (4.29) is described below:

\[
\frac{d[A]}{dt} = k_1[A][B][C][D] - k_2[E][F] 
\]  
(4.30)

\[
\frac{d[B]}{dt} = k_1[A][B][C][D] - k_2[E][F] 
\]  
(4.31)

\[
\frac{d[C]}{dt} = k_1[A][B][C][D] - k_2[E][F] 
\]  
(4.32)

\[
\frac{d[D]}{dt} = k_1[A][B][C][D] - k_2[E][F] 
\]  
(4.33)

\[
\frac{d[E]}{dt} = -k_1[A][B][C][D] + k_2[E][F] 
\]  
(4.34)

\[
\frac{d[F]}{dt} = -k_1[A][B][C][D] + k_2[E][F] 
\]  
(4.35)

Based on the equations (4.30) to (4.35), the ODE models for the reactions 9 and 10 can be obtained as:

\[
\frac{d[ipromI]}{dt} = k_9[luxbox][LrA\textsubscript{i2}][promI][promR] - k_{10}[ipromI][ipromR] 
\]  
(4.36)

\[
\frac{d[ipromR]}{dt} = k_9[luxbox][LrA\textsubscript{i2}][promI][promR] - k_{10}[ipromI][ipromR] 
\]  
(4.37)

\[
\frac{d[luxbox]}{dt} = -k_9[luxbox][LrA\textsubscript{i2}][promI][promR] + k_{10}[ipromI][ipromR] 
\]  
(4.38)
\[ \text{d}[\text{LrA}_{12}]/\text{dt} = -k_9.[\text{luxbox}][\text{LrA}_{12}][\text{promI}][\text{promR}] + k_{10}.[\text{ipromI}][\text{ipromR}] \]  \hspace{1cm} (4.39) \\
\[ \text{d}[\text{promI}]/\text{dt} = -k_9.[\text{luxbox}][\text{LrA}_{12}][\text{promI}][\text{promR}] + k_{10}.[\text{ipromI}][\text{ipromR}] \]  \hspace{1cm} (4.40) \\
\[ \text{d}[\text{ipromR}]/\text{dt} = -k_9.[\text{luxbox}][\text{LrA}_{12}][\text{promI}][\text{promR}] + k_{10}.[\text{ipromI}][\text{ipromR}] \]  \hspace{1cm} (4.41) \\

**Reaction 11:** 
\[ k_{11} \]
\[ \text{promR} \rightarrow \text{promR} + \text{mRNAR} \]  \hspace{1cm} (4.42)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 11 is similar to the equation (4.43).

\[ k_j \]
\[ \text{A} \rightarrow \text{A} + \text{B} \]  \hspace{1cm} (4.43)

The ODE template for the equation (4.43) is given as:

\[ \text{d}[\text{B}]/\text{dt} = k_j.[\text{A}] \]  \hspace{1cm} (4.44)

Based on the equation (4.44), the ODE model for the reaction 11 can be obtained as:

\[ \text{d}[\text{mRNAR}]/\text{dt} = k_{11}.[\text{promR}] \]  \hspace{1cm} (4.45)

**Reaction 12:** 
\[ k_{12} \]
\[ \text{promI} \rightarrow \text{promI} + \text{mRNAI} \]  \hspace{1cm} (4.46)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 12 is similar to the equation (4.43).

Based on the equation (4.44), the ODE model for the reaction 12 can be obtained as:

\[ \text{d}[\text{mRNAI}]/\text{dt} = k_{12}.[\text{promI}] \]  \hspace{1cm} (4.47)
Reaction 13: \[ k_{13} \]
\[ \text{ipromR} \rightarrow \text{ipromR} + \text{mRNAR} \]  \hspace{1cm} (4.48)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 13 is similar to the equation (4.43).

Based on the equation (4.44), the ODE model for the reaction 13 can be obtained as:

\[
d[\text{mRNAR}]/dt = k_{13} \cdot [\text{ipromR}] \hspace{1cm} (4.49)
\]

Reaction 14: \[ k_{14} \]
\[ \text{ipromI} \rightarrow \text{ipromI} + \text{mRNAI} \]  \hspace{1cm} (4.50)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 14 is similar to the equation (4.43).

Based on the equation (4.44), the ODE model for the reaction 14 can be obtained as:

\[
d[\text{mRNAR}]/dt = k_{14} \cdot [\text{ipromI}] \hspace{1cm} (4.51)
\]

Reaction 15: \[ k_{15} \]
\[ \text{mRNAR} \rightarrow \text{mRNAR} + \text{LuxR} \]  \hspace{1cm} (4.52)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 15 is similar to the equation (4.43).

Based on the equation (4.44), the ODE model for the reaction 15 can be obtained as:

\[
d[\text{LuxR}]/dt = k_{15} \cdot [\text{mRNAR}] \hspace{1cm} (4.53)\]
Reactions:

**Reaction 16:**

\[ k_{16} \]

\[ \text{mRNAI} \rightarrow \text{mRNAI} + \text{LuxI} + \text{LuxA/B} \]  

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-two output system. The reaction 16 is similar to the equation (4.55).

\[ k_i \]

\[ A \rightarrow A + B + C \]  

(4.55)

The ODE template for the equation (4.55) is given as:

\[ \frac{d[B]}{dt} = k_i[A] \]  

(4.56)

\[ \frac{d[C]}{dt} = k_i[A] \]  

(4.57)

Based on the equations (4.56) and (4.57), the ODE model for the reaction 16 can be obtained as:

\[ \frac{d[\text{LuxI}]}{dt} = k_{16}[\text{mRNAI}] \]  

(4.58)

\[ \frac{d[\text{LuxA/B}]}{dt} = k_{16}[\text{mRNAI}] \]  

(4.59)

**Reaction 17:**

\[ k_{17} \]

\[ \text{mRNAR} \rightarrow \text{mRNAR}_{\text{decay}} \]  

(4.60)

The ODE template for the reaction 17 can be obtained by considering the system to be a decay system. The reaction 17 is similar to the equation (4.60) as described below:

\[ k_i \]

\[ A \rightarrow \text{decay} \]  

(4.60)

The ODE template for the equation (4.60) is given as:

\[ \frac{d[A]}{dt} = -k_i[A] \]  

(4.61)
Based on equation (4.61), the ODE model for the reaction 17 can be described as:

\[
\frac{d[mRNAR]}{dt} = -k_{17}[mRNAR] \quad (4.62)
\]

**Reaction 18:**

\[
k_{18} \quad \text{mRNAI} \rightarrow \text{mRNAI}_{\text{decay}}
\]

\[
(4.63)
\]

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 17 is similar to the equation (4.61). The ODE template for the equation (4.61) is given by equation (4.62).

Based on equation (4.62), the ODE model for the reaction 18 can be described as:

\[
\frac{d[mRNAI]}{dt} = -k_{18}[mRNAI]
\]

**Reaction 19:**

\[
k_{19} \quad \text{LuxI} \rightarrow \text{LuxI}_{\text{decay}}
\]

\[
(4.64)
\]

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 19 is similar to the equation (4.61). The ODE template for the equation (4.61) is given by equation (4.62).

Based on equation (4.61), the ODE model for the reaction 19 can be described as:

\[
\frac{d[\text{LuxI}]}{dt} = -k_{19}[\text{LuxI}]
\]

**Reaction 20:**

\[
k_{20} \quad \text{LuxR} \rightarrow \text{LuxR}_{\text{decay}}
\]

\[
(4.66)
\]

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 20 is similar to the equation (4.61). The ODE template for the equation (4.61) is given by equation (4.62).
Based on equation (4.62), the ODE model for the reaction 20 can be described as:

\[
\frac{d[\text{LuxR}]}{dt} = -k_{20} [\text{LuxR}] 
\]

(4.67)

**Reaction 21:**

\[ k_{21} \]

\[ \text{LuxI} \rightarrow \text{LuxI} + A_i \]  

(4.68)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 21 is similar to the equation (4.43).

Based on the equation (4.44), the ODE model for the reaction 21 can be obtained as:

\[
\frac{d[A_i]}{dt} = k_{21} [\text{LuxI}] 
\]

(4.69)

**Reaction 22:**

\[ k_{22} \]

\[ A_i \rightarrow A_{\text{idecay}} \]  

(4.70)

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 22 is similar to the equation (4.61). The ODE template for the equation (4.61) is given by equation (4.62).

Based on equation (4.62), the ODE model for the reaction 22 can be described as:

\[
\frac{d[A_i]}{dt} = -k_{22} [A_i] 
\]

(4.71)

**Reaction 23:**

\[ k_{23} \]

\[ A_{\text{input}} \rightarrow A_i + A_{\text{input}} \]  

(4.72)

The ODE templates for the above reactions are obtained by considering the reactions to be a one input-one output system. The reaction 23 is similar to the equation (4.43).
Based on the equation (4.44), the ODE model for the reaction 23 can be obtained as:

\[
\frac{d[A_i]}{dt} = k_{23}.[A_{\text{input}}] \tag{4.73}
\]

**Reaction 24:**

\[
k_{24} \quad \text{LuxA/B} \rightarrow \text{LuxA/B}_{\text{decay}} \tag{4.74}
\]

The ODE template for the above reaction can be obtained by considering the system to be a decay system. The reaction 24 is similar to the equation (4.61). The ODE template for the equation (4.61) is given by equation (4.62).

Based on equation (4.62), the ODE model for the reaction 24 can be described as:

\[
\frac{d[\text{LuxA/B}]}{dt} = -k_{24}.[\text{LuxA/B}] \tag{4.75}
\]

### 4.2.1 Mathematical simulation of the Bioluminescence system

From the previous section, it has been observed that the ODE model of the Bioluminescence system consists of 24 biochemical reactions, 24 rate constants and 17 species. These species are \(A_i\), LuxR, LrA_i, LrA_i2, LuxD, LuxD_{complex}, ipromR, DNA_loop, mRNAR, mRNAI, LuxI, A_{isource}, Luxbox, promI, promR, ipromI and LuxA/B. The concentration of \(A_{isource}\), which is the input to the cell, is fixed at 10µM to show the activation of luminescence activity by the auto-inducer. This input is then reduced to 0 µM, to show the inhibition of luminescence activity. The ipromR concentration is fixed at 10 μM when \(A_{isource}\) is 10 µM and 1 μM when \(A_{isource}\) is 0. This ODE model described in our research is simulated using Matlab ODE solver (ode45 and ode113). The input to the V. Fischeri cell is the auto-inducer, \(A_i\) and the output of the system is the luminescence (LuxA/B) produced.
<table>
<thead>
<tr>
<th>Species</th>
<th>A&lt;sub&gt;i&lt;/sub&gt;</th>
<th>LuxR</th>
<th>LrA&lt;sub&gt;i&lt;/sub&gt;</th>
<th>LrA&lt;sub&gt;i2&lt;/sub&gt;</th>
<th>LuxD</th>
<th>LuxD&lt;sub&gt;complex&lt;/sub&gt;</th>
<th>ipromR</th>
<th>DNA&lt;sub&gt;loop&lt;/sub&gt;</th>
<th>mRNA&lt;sub&gt;R&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conc.</td>
<td>1</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10/1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Species</th>
<th>mRNA&lt;sub&gt;I&lt;/sub&gt;</th>
<th>LuxI</th>
<th>A&lt;sub&gt;isource&lt;/sub&gt;</th>
<th>Luxbox</th>
<th>prom&lt;sub&gt;I&lt;/sub&gt;</th>
<th>promR</th>
<th>iprom&lt;sub&gt;I&lt;/sub&gt;</th>
<th>LuxA/B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conc.</td>
<td>0</td>
<td>0</td>
<td>10/0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1 Biochemical species and the initial concentrations of the ODE model of the bioluminescence system [9]

<table>
<thead>
<tr>
<th>Rate</th>
<th>k&lt;sub&gt;1&lt;/sub&gt;</th>
<th>k&lt;sub&gt;2&lt;/sub&gt;</th>
<th>k&lt;sub&gt;3&lt;/sub&gt;</th>
<th>k&lt;sub&gt;4&lt;/sub&gt;</th>
<th>k&lt;sub&gt;5&lt;/sub&gt;</th>
<th>k&lt;sub&gt;6&lt;/sub&gt;</th>
<th>k&lt;sub&gt;7&lt;/sub&gt;</th>
<th>k&lt;sub&gt;8&lt;/sub&gt;</th>
<th>k&lt;sub&gt;9&lt;/sub&gt;</th>
<th>k&lt;sub&gt;10&lt;/sub&gt;</th>
<th>k&lt;sub&gt;11&lt;/sub&gt;</th>
<th>k&lt;sub&gt;12&lt;/sub&gt;</th>
<th>k&lt;sub&gt;13&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>0.1</td>
<td>2.0</td>
<td>0.06</td>
<td>4.0</td>
<td>0.01</td>
<td>3.0</td>
<td>6.0</td>
<td>1.0</td>
<td>0.1</td>
<td>5.0</td>
<td>0.01</td>
<td>0.00000015</td>
<td>0.06</td>
</tr>
</tbody>
</table>

Table 4.2 Rate constants for the bioluminescence system [9]

The Bioluminescence system is modeled mathematically by using the ODE’s. R. Krishnan [9] had used the ode15s solver to solve the ordinary differential equations developed for the biochemical reactions of the Bioluminescence system. However, ode45 should be the first ordinary differential equation solver that is used while solving a set of ordinary differential equations. The ode113 is another ode solver that is used because of its high accuracy. The ODE’s solve the differential equations at certain fixed intervals of time, determined by the time step specified. The ode45 and ode113 though used for non-stiff equations are very accurate when compared to the ode15s solver. Stiffness is actually an efficiency issue and if we are not concerned with the computation time, we then would not be concerned with stiffness too.
4.2.2 Stochastic modeling of the Bioluminescence system

The previous section described how the ordinary differential equations formed for the biochemical reactions are solved using the ODE solver (ode45 and ode113). However, the ODE model has its own disadvantages as it does not take into account the fluctuations that occur in the biological systems and assumes that the process is entirely deterministic. Therefore, the stochastic model was developed to study the robustness of the system in the presence of noise and also the random behavior of the biochemical reactions. R. Krishnan [9] had developed a stochastic model to simulate the bioluminescence system based on the Gillespie Algorithm. The Gillespie Algorithm mainly answers two questions.

a) The time (τ) when the next reactions fires

b) The reaction (reaction channel index μ) that will fire next [14].

In our research, we modeled the bioluminescence system by using Gibson and Bruck’s Next Reaction Method which is advancement over the Gillespie Algorithm. The Gillespie Algorithm used by R. Krishnan [9] and Gibson and Bruck’s Next Reaction method implemented in our research is described briefly in the subsequent sections.

4.3 Gillespie’s Algorithm for stochastic modeling

Gillespie developed a First Reaction Method, which generates a putative time τᵢ for each reaction to take place. By putative time, we refer to the time the reaction would occur if no other reaction had occurred first. Then, let μ be the reaction whose putative time is first and its putative time is τᵢμ. The algorithm for Gillespie’s First Reaction Method is described in the next page:
1) Initialization (i.e., set the initial number of molecules, set $t \leftarrow 0$, time of simulation is set $t_{\text{initial}}$ and $t_{\text{final}}$). The stochastic reaction rates ($c_1, c_2, \ldots$) are calculated for all the reactions, the reaction rates are stochastic.

2) Calculation of the propensity function, $a_i$, for all $i$ as given in [12].

3) For each value of $i$, a putative time, $\tau_i$, is being generated based on the exponential distribution with parameter $a_i$.

4) Let $\mu$ be the reaction whose putative time $\tau_\mu$ is least.

5) Change the number of molecules so as to reflect the execution of the reaction $\mu$. Also, set $t \leftarrow t + \tau_\mu$.

6) Go to step 2. Repeat the steps until $t = t_{\text{final}}$.

This algorithm basically uses ‘$r$’ random iterations where ‘$r$’ denotes the number of the reactions, takes time proportional to ‘$r$’ to update the propensities ‘$a_i$’ and also takes time proportional to ‘$r$’ to identify the smallest $\tau_\mu$ [14].

Two modifications were performed on this algorithm when implementing it for solving the Bioluminescence system.

a) The value of the reaction channel index $\mu$ was chosen by generating a random number from 1 to 24 which denotes the total number of the biochemical reactions involved in this system. This value then decides the type of the biochemical reaction ($R_\mu$) that would occur at the discrete time interval, $t_{\text{current}} + \tau$.  

60
b) Also, the biomolecular species in the reaction $R_{\mu}$ are adjusted based on the reaction. The reactants in the reaction are decreased and the products are increased [9].

### 4.4 Gibson and Bruck’s Next Reaction Method

Gibson and Bruck’s Next Reaction Method is an extension to Gillespie’s Algorithm. The following steps take place at very iteration of the Gillespie Algorithm and take time proportional to the number of reactions ‘r’.

a) Updating all the reaction (‘r’) propensities $a_i$.

b) Generating a putative time $\tau_i$, for each and every ‘i’; and

c) Identifying the smallest putative time, $\tau_{\mu}$.

Gibson and Bruck’s Next Reaction Method will avoid carrying out each of the above activities in turn. With the Next Reaction Method, instead of computing the time taken for each reaction, one actually deals with the time at which a reaction occurs and unlike in the Gillespie Algorithm, these times are not computed afresh at each step, but may be stored and then re-used. These reaction times are stored in an indexed priority queue. Further, in Gibson and Bruck’s Next Reaction method, the propensities are computed only in case they have changed. The algorithm for the next reaction method is as described below:

1) Initialization:

a) Set the initial number of molecules, set $t \leftarrow 0$ and time of simulation is set to $t_{\text{initial}}$. The Stochastic reaction rates ($c_1$, $c_2$ …….) are calculated for all the reactions, the reaction rates are stochastic. Generate a dependency graph $G$. 
b) Calculation of the propensity function, $a_i$, for all ‘i’ as given in [12].

c) For each value of i, a putative time, $\tau_i$, is being generated based on the exponential distribution with parameter $a_i$.

d) Store the $\tau_i$ values in an indexed priority queue, $\rho$.

2) Let $\mu$ be the reaction whose putative time $\tau_\mu$ stored in the priority index $\rho$ is least.

3) Let $\tau$ be $\tau_\mu$.

4) Change the number of molecules so as to reflect the execution of the reaction $\mu$. Also, set $t \leftarrow \tau$.

5) For each edge $(\mu, \alpha)$ in the dependency graph $G$:
   
a) $a_\alpha$ is updated.
   
b) If $\alpha \neq \mu$, set $\tau_\alpha \leftarrow (a_{\alpha, new} / a_{\alpha, old}) (\tau_\alpha - t) + t$
   
c) In case $\alpha = \mu$, then a random number, $\nu$, is generated according to an exponential distribution with parameter $a_\mu$, and set $\tau_\alpha = \nu + t$;
   
d) Replace the old value $\tau_\alpha$ in indexed priority queue, $\rho$, with the new value.

6) Go to step 2 and repeat the process.

Figure 4.5 below represents the one step simulation flow of the Gillespie Algorithm, Gibson and Bruck’s Next Reaction Method and the Tau-Leaping Method.
4.4.1 Dependency graph

The dependency graph is basically a data structure that precisely mentions as which of the propensity functions \( a_i \)s should change when a given reaction is executed. Each of the reaction channels \( \mu \) is denoted as a node in the graph. A directed edge connects reaction \( R_i \) to reaction \( R_j \) if and only if the reaction \( R_i \) affects the reactants in the reaction \( R_j \). Therefore, in our research, the dependency graph is used to recalculate the minimal number of propensity functions in step 5 [14]. The dependency graph for the Phage Lambda system is shown in Figure 4.6.
4.4.2 Indexed priority queue

For most applications, the binary heap is a very efficient means of implementing a priority queue. For heap ‘H’ with ‘N’ elements, one can access the minimum or the least element in constant time. Every insertion or deletion of an element from a heap takes $O \log (N)$ time for a total running time of $O (N \log N)$. Though binary heap is rarely considered to be the most efficient data structure for a particular application, it is usually efficient enough.

The simplest method of implementing a priority queue is to store the elements into an array and then use a linear search technique to find the minimum element. The time taken to compute the minimum element using the linear search is $O (N)$ where ‘N’ represents the number of elements. The insertion, deletion and modification of the elements are usually done in a constant time. With the Next Reaction method, the linear search is the most efficient algorithm especially when the number of reactions is small [67].
5 Experiments and Results

In this chapter, the different experiments that are carried out in this research are explained in detail and all the results associated with the experiments are plotted and showed in a graphical representation.

5.1 Deterministic simulation of the Phage Lambda system

As discussed in chapter 3, the Phage Lambda system behaves as a bio-inverter. The bio-inverter consists of eight different species, $R_p$, $R_{p2}$, $C_g$, $C_gR_{p2}$, Cro, mRNA, pRNA and rRNA. Here, while carrying out the deterministic simulation of the Phage Lambda System, we assume the concentrations of pRNA and rRNA as 1Molar and the concentration of $C_g$ as 0.07µM. The bio-inverter model contains ten reactions and also has ten reaction constants associated with them. The rate constants are obtained from Weiss and Ptashne [56, 57] and are denoted $k_1$ to $k_{10}$. The rate constants that have been used for modeling the Phage Lambda system were already mentioned in chapter 3 and are reproduced here.

<table>
<thead>
<tr>
<th>Rate</th>
<th>$K_1$</th>
<th>$K_2$</th>
<th>$K_3$</th>
<th>$K_4$</th>
<th>$K_5$</th>
<th>$K_6$</th>
<th>$K_7$</th>
<th>$K_8$</th>
<th>$K_9$</th>
<th>$K_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>8.333</td>
<td>0.1667</td>
<td>0.5775</td>
<td>0.5775</td>
<td>66.67</td>
<td>0.2</td>
<td>0.0001</td>
<td>0.03</td>
<td>2.0</td>
<td>0.5575</td>
</tr>
</tbody>
</table>

Table 5.1 Rate constants for modeling the Phage Lambda system [9]

In order to maintain a constant supply of the input protein ($R_p$), a “drive” term is added to the rate equation that involves the input protein ($R_p$) similar to the Weiss model [53]. The bio-inverter model was created and simulated using Matlab (MATLAB 7.8.0 (R2009a)) on an Intel(R) Core™2 Duo CPU, 3.16 GHz, and 3.23GB of RAM.
The deterministic simulation of the Phage Lambda system was carried out using the ode45 and ode113 solvers. The ode45 is the first solver that is to be tried while solving a set of differential equations. However, the ode45 and ode113 are used for solving non-stiff equations mostly and while solving stiff equations ode45 and ode113 can still be used but the speed reduces considerably. This can be taken care of sometimes by introducing the options term in the syntax of the ode solver. The Figure 5.1 below shows the transient characteristics of the concentrations of the input protein ($R_p$) and the output protein (Cro) in $\mu$M with respect to the time.

![Transient characteristics of the Phage Lambda system](image)

**Figure 5.1 Transient characteristics of the phage lambda inverter using ode45 solver (Matlab)**

The concentrations of the input protein ($R_p$) and the output protein (Cro) are plotted against the time in seconds with a time span ranging from “0” to “2000” represented in Matlab as [0 2000]. From the plot it is observed that the characteristics are quite similar to that of an electrical inverter. The slight overshoot observed in the $R_p$ concentration from the plot is due to the external drive that was added to the system. This drive helps maintain a constant supply of input
protein for a given time period. The inverter has a low transition time which is mainly attributed to the high repressor affinity. The dynamic behavior of the bio-inverter is similar and hence comparable to that of an electrical inverter. In Figure 5.1, the x-axis represents the time in seconds and the y-axis represents the concentrations of the input protein (Rp) and the output protein (Cro) in µM. The solid blue line in the figure represents the Rp concentration and the dotted red line represents the Cro concentration. The Cro concentration is zero throughout the time period because the value of Cro concentration is very low and of the order of 10^{-6}. The scale of the plot is not sufficient enough to represent both the Rp and Cro concentrations accurately. The individual concentrations of Rp and Cro for the time intervals ranging from [0 500], [500 1000], [1000 1500] and [1500 2000] are shown in the Figures 5.2 to 5.10.

![Graph of Rp vs Cro concentration](image)

**Figure 5.2** Transient characteristics of the Phage Lambda system in the time range [0 500] using ode45 solver (Matlab)
Figure 5.3 Transient characteristics of the Phage Lambda system in the time range [500 1000] using ode45 solver (Matlab)

Figure 5.4 Transient characteristics of the Phage Lambda system in the time range [1000 1500] using ode45 solver (Matlab)
Next, the deterministic simulation of the Phage Lambda system is carried out using the ode113 solvers. As already discussed in chapter 2, ode113 is a Matlab solver that is mainly used for non-stiff equations but can be preferred even for stiff equations because of its high accuracy. Figures 5.6 to 5.10 show the transient characteristics of the Phage Lambda system using ode113 solver over the time range [0 2000] and also for the time ranges [0 500], [500 1000], [1000 1500] and [1500 2000] individually. The results obtained by simulating the Phage Lambda system using ode45 and ode113 are then compared with the results of R. Krishnan [9].
Figure 5.6 Transient characteristics of the Phage Lambda system using ode113 solver (Matlab)

Figure 5.7 Transient characteristics of the Phage Lambda system in the time range [0 500] using ode113 solver (Matlab)
Figure 5.8 Transient characteristics of the Phage Lambda system in the time range [500 1000] using ode113 solver (Matlab)

Figure 5.9 Transient characteristics of the Phage Lambda system in the time range [1000 1500] using ode113 solver (Matlab)
Figure 5.10 Transient characteristics of the Phage Lambda system in the time range [1500 2000] using ode113 solver (Matlab)

Figure 5.11 shows the transient characteristics of the Phage Lambda system that was simulated in Matlab using the ode15s solver by R. Krishnan [9] in his thesis. From Figure 5.11 we can observe that the transient characteristics of the Phage Lambda system obtained by simulating the system using the Matlab ode solvers ode45 and ode113 are similar to the transient characteristics of the Phage Lambda system using the Matlab ode solver ode15s. The results we have obtained using ode45 and ode113 are accurate when compared to the ones obtained using ode15s.
5.2 Implementation of the stochastic simulation method

Several environments and software tools have been developed to carry out the stochastic simulation of biological systems. Some of the software tools that perform the stochastic simulation of the biological systems include Cain [42], Synbioss [43], SBML [44], Dizzy [45], FERN [46], stochastic simulation using MATLAB [47], and STOCKS [48]. Some of these software tools involve complex object oriented programming. After careful investigation and study we have decided to perform the stochastic simulation of the biological pathways by writing a code in Python using the Gibson and Bruck’s Next Reaction Method. The following section describes in brief about the Python programming language.
5.2.1 Python programming language

Python is a general purpose high-level programming language. Python supports multi programming paradigms which primarily includes object oriented, imperative and functional and it also features a fully dynamic type system and includes automatic memory management similar to the Perl, Ruby and Scheme programming languages. Python is used as a scripting language like several other dynamic languages. Python was developed in the late 1980s and was first implemented in December 1989 by its designer Guido van Rossum at the National Research Institute for Mathematics and Computer Science as a successor to the ABC programming language. The Python programming language, rather than forcing programmers to follow a particular style of programming, permits several styles such as object oriented programming and the structured programming. For the memory management, Python uses dynamic typing and a combination of reference counting and a cycle detecting garbage collector. An important feature of Python that binds the method and the variable names during program execution is the name resolution. Python is frequently used as a scripting language for several web applications. Python has been extensively used in the information security industry. For several operating systems like the Linux distributions, NetBSD, OpenBSD and Mac OS X, Python is a standard component. Python was intended to be a highly readable language and is designed in such a way that it has a clean layout using mostly English keywords where other programming languages relied heavily on punctuations. Another important feature of Python is the indentation. Instead of curly braces or keywords to delimit the blocks, Python uses whitespace indentation. After certain statements, we come across the increase in indentation in Python programming language. The decrease in indentation signifies the end of a current block.
The Python’s statements include the following:

1. The **if statement** which executes a block of code conditionally along with else and elif (which is a contraction of else-if).

2. The **for statement** which iterates over an iterable object thereby capturing each element to the local variable.

3. The **while statement** which executes a block of code as long as a condition is true.

4. The **try statement** which allows errors thrown in its attached code to be caught and handled except clauses.

5. The **class statement** which executes a block of code and attaches the local namespace to a class to be used in object oriented programming.

6. The **def statement** which defines a function.

Python uses the concept called duck typing which is a style of dynamic typing in which an object’s current set of methods and properties determines the valid semantics instead of inheriting from a particular class or implementing a specific interface. Python has typed objects but untyped variable names. Python also allows programmers to define their own types using classes and these types are most often used for object oriented programming. The mainstream Python implementation is written in the C programming language and is termed CPython. This CPython compiles the Python program into intermediate bytecode which is then executed by the virtual machine. The CPython is distributed with a standard library written in a mixture of C and Python. There is also another implementation of the Python programming language known as the Jython which compiles the Python program into the Java byte code and this is executed by
every Java virtual machine implementation. Thereby, this enables the usage of Java class library functions from the Python program. One of the greatest strengths of the Python programming languages is the presence of a large standard library which provides pre written tools for many tasks [68].

5.2.2 Generalized template for the Python code used in the research

In this section, we describe a generalized template for the Python code that was used in the research for simulating the Phage Lambda system and the Bioluminescence system based on Gibson and Bruck’s Next Reaction Method.

The template consists of two Python files, gillespie_helper.py and nextrxn.py. The gillespie_helper.py is a Python file that provides helper function for the Gillespie implementation. The nextrxn.py is a Python code that implements Gibson and Bruck’s Next Reaction Method.

To run the Next Reaction Method in nextrxn.py, first go to “Start” menu on the desktop of your computer. Click on the “Run” option and type “cmd” and click ok. This takes you to the command window. Then go to the respective directory where your files is stored and then type “<nextrxn.py><switch.rxns><switch.spcs><output file name><# of output time points>”.

The switch.rxns represents the reactions of the biological system which is simulated using the Next Reaction Method and the switch.spcs represents the species and the initial number of molecules of the species described.

The screenshots for the templates gillespie_helper.py, nextrxn.py and the command window from which the Gibson and Bruck’s Next Reaction Method is executed are shown in Figures 5.12, 5.13 and 5.14 respectively. Figures 5.15 and 5.16 show the screenshots for the input
arguments switchphagelambda.rxns and switchphagelambda.spcs which represent the reactions and the initial number of molecules of the species of the Phage Lambda system.

```python
# helper functions for the gillespie implementation.

# populate dictionaries from input file
def ReadInputFile(input_filename):
    substrates_dict = {}
    products_dict = {}
    c_dict = {}

    input_file = open(input_filename, "r")
    line_count = 0
    cur_cxn = None

    for line in input_file:
        this_line = line.strip()
        speci = line_count % 5
        args = this_line.split()

        if len(args) != 3:
            line_count += 1
        continue

        ltyp = args[0]
        rem = args[1:]

        if ltyp == "Rxn":
            our_cxn = rem[0]
        elif ltyp == "Sub":
            substrates_dict[our_cxn] = ()
            for species in rem:
                if species in substrates_dict[our_cxn]:
                    substrates_dict[our_cxn][species] += 1
                else:
                    substrates_dict[our_cxn][species] = 1
        elif ltyp == "Prod":
            products_dict[our_cxn] = ()
            for species in rem:
                if species in products_dict[our_cxn]:
                    products_dict[our_cxn][species] += 1
                else:
                    products_dict[our_cxn][species] = 1
        elif ltyp == "Rate":
            c_dict[our_cxn] = float(rem[0])

    # will ignore species
    line_count += 1
```

Figure 5.12 Screenshot showing the gillespie_helper.py template
Figure 5.13 Screenshot showing the nextrxn.py template which describes Gibson and Bruck’s Next Reaction Method

Figure 5.14 Screenshot of the command window from which the Next Reaction Method is executed
Figure 5.15 Screenshot of the text file describing the reactions of the Phage Lambda system

Figure 5.16 Screenshot of the text file showing the initial number of molecules of the Phage Lambda system
5.3 Stochastic simulation of the Phage Lambda system

The ODE’s are solved using the Matlab ode solvers ode45 and ode113. The ode solver solves the differential equations at certain fixed intervals of time which are determined by the time step specified. The major drawback of the ODE model is that the ODE model of any biological pathway assumes that the process is deterministic without any fluctuations in the system. Hence, the ODE model cannot clearly describe the fluctuations and instabilities that occur in the biological system at the molecular level.

In a biological system, the population at the molecular level is not deterministic and hence some amount of randomness or noise is to be incorporated to accurately determine the results from the simulation. In our research a stochastic model of the Phage Lambda system was developed by using Gibson and Bruck’s Next Reaction method and this algorithm for the Phage Lambda system was implemented using the Python programming language. As already explained in the earlier sections, the Phage Lambda system exhibits inverter characteristics. The stochastic model was simulated to verify the inverter characteristics and also to verify the robustness of the bio-inverter in the presence of some noise and also the random order of the biological reactions.

The Lambda lysis-lysogeny pathway has already been explained by McAdams and Arkin [61]. Their work includes the models of the promoter sites and operator and produces a complex and complete model of the lambda lysis-lysogeny pathway. However, the modeling and simulation of the lambda lysis-lysogeny pathway is complex and difficult and hence we considered a simplified model of the bio-inverter from a circuit’s perspective in our research. The bio-inverter model is described by ten biochemical reactions (R₁, R₂,…..R₁₀) and these reactions are controlled by reaction rates (c₁, c₂,…..c₁₀) which are probabilities per unit time. The volume was
fixed at $10^{-3}$ liters. The simulation was run for a time period of 2000 seconds to show the Lytic and Lysogenic states of the Phage Lambda system.

The Phage Lambda system exhibits two states. When the input $R_{p2}$ is high, then it represents the Lysogeny (OFF) state and when the output Cro is high, then it represents the Lytic (ON) state. The stochastic model was simulated using the parameters described in the Tables 5.2 and 5.3. These are same as the values in [9].

<table>
<thead>
<tr>
<th>Species</th>
<th>$R_p$</th>
<th>$R_{p2}$</th>
<th>$C_g$</th>
<th>$C_g R_{p2}$</th>
<th>pRNA</th>
<th>rRNA</th>
<th>mRNA</th>
<th>Cro</th>
</tr>
</thead>
<tbody>
<tr>
<td>Molecules</td>
<td>50</td>
<td>1</td>
<td>30</td>
<td>1</td>
<td>20</td>
<td>20</td>
<td>1</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 5.2 Initial number of molecules for the biochemical species for the “ON” or Lytic state

<table>
<thead>
<tr>
<th>Species</th>
<th>$R_p$</th>
<th>$R_{p2}$</th>
<th>$C_g$</th>
<th>$C_g R_{p2}$</th>
<th>pRNA</th>
<th>rRNA</th>
<th>mRNA</th>
<th>Cro</th>
</tr>
</thead>
<tbody>
<tr>
<td>Molecules</td>
<td>250</td>
<td>1</td>
<td>30</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5.3 Initial number of molecules for the biochemical species for the “OFF” or Lysogenic state

![Stochastic simulation of the “ON” state or the Lytic State of the Phage Lambda System](image)
From Figure 5.18, we can verify that the Phage Lambda system switches to the Lytic state or the “ON” state and similarly from Figure 5.19, we can verify that the Phage Lambda system switches to the Lysogenic state or the “OFF” state. The corresponding figures for the Gillespie method from [7] are shown in Figures 5.20 and 5.21.
Figure 5.20 Stochastic simulation of the “OFF” state or the Lysogenic state of the Phage Lambda system using the Gillespie Algorithm. Reprinted from [9].

The Figure 5.22 shown below represents the plot obtained for the results of the ABM of the Phage Lambda system.

Figure 5.21 Simulation of the Phage Lambda system using Agent Based Modeling. Reprinted from [13].
From Figures 5.18 and 5.19 we can observe that the Phage Lambda system exhibits some inverter characteristics by toggling between the “ON” and “OFF” states and hence it is considered to be a bio-inverter whose characteristics is quite similar to that of an electrical inverter. The curves for the stochastic simulation are similar to those from [9] and [13].

5.4 Deterministic simulation of the Bioluminescence system

The ODE model and the rate constants for the bioluminescence system that were used by R. Krishnan [9] were actually extracted from Cox et al. [65]. There are a total of seventeen species in the bioluminescence system and the entire system is represented by 24 reactions and 24 rate constants. The parameters for the bioluminescence system are already described in chapter 4 and are reproduced here.

<table>
<thead>
<tr>
<th>Species</th>
<th>Ai</th>
<th>LuxR</th>
<th>LrAi</th>
<th>LrAi2</th>
<th>LuxD</th>
<th>LuxD_complex</th>
<th>ipromR</th>
<th>DNA_loop</th>
<th>mRNA_R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conc.</td>
<td>1</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10/1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Species</th>
<th>mRNAI</th>
<th>LuxI</th>
<th>Aisource</th>
<th>Luxbox</th>
<th>promI</th>
<th>promR</th>
<th>ipromI</th>
<th>LuxA/B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conc.</td>
<td>0</td>
<td>0</td>
<td>10/0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.4 Biochemical species and the initial concentrations of the ODE model of the bioluminescence system [9]
Table 5.4 describes the number of molecules of each of the species and Table 5.5 describes the rate constants that are used for each of the 24 reactions involved in the bioluminescence system. The $A_{isource}$ or the input to the cell is fixed at 10 µM to show the activation of the luminescence activity by the auto-inducer. The $A_{isource}$ is then reduced to 0 µM to show the inhibition of the luminescence activity. The concentration of the ipromR is fixed at 10 µM when the value of the $A_{isource}$ is 10 µM and the concentration of the ipromR is fixed at 1 µM when the value of the $A_{isource}$ is 0 µM. The ODE model was simulated using the ode solvers (ode45 and ode113). The input to the system ($V. Fischeri$) is the auto-inducer, $A_i$. The output of the system is measured in terms of the luminescence (Lux A/B) produced. The ODE model was simulated for a period of 1500 seconds and the interval was maintained at 5 seconds in order to avoid integration errors due to small steps. The output when the input $A_{isource}$ is 0 µM is shown in the Figure 5.23 and the output when the input $A_{isource}$ is 10 µM is shown in the Figure 5.24.
Simulation Results of Bioluminescence System with $A_{source} = 0$

Figure 5.22 Simulation results of the bioluminescence system using the Matlab ode45 solver with the input $A_{source} = 0$

Simulation Results of Bioluminescence System with $A_{source} = 10$

Figure 5.23 Simulation results of the bioluminescence system using the Matlab ode45 solver with the input $A_{source} = 10$
In Figure 5.23, the solid blue line represents the concentration of the input to the system, the auto-inducer ($A_i$), the dotted red line represents the concentration of LuxI and the solid green line represents the concentration of the output of the system or the luminescence LuxA/B. From this figure, we can observe that the concentrations of the auto inducer $A_i$ and the LuxI are low and hence there is no production of light which is represented by the output LuxA/B.

In Figure 5.24 the representations of the concentrations remain the same as the Figure 5.20 with an addition of the concentration of the LuxR protein which is represented by the solid black line. In this figure, we can observe that the concentrations of the auto inducer $A_i$ is high and hence there is production of light which is represented by the output LuxA/B. As $A_{source}$ is introduced into the cell, the concentration of the output of the system LuxA/B increases considerably and at 1500 seconds we can observe that the concentration of the output LuxA/B is around 2.58µM. The LuxA/B concentration keeps increasing till the end of the simulation since the concentration of the $A_{source}$ is always high (10µM).

Figures 5.25 and 5.26 represent the simulation of the Bioluminescence system using the ode113 solver with the inputs $A_{source} = 0$ and $A_{source} = 10$. These plots are then compared to the results obtained by R. Krishnan [9] in his thesis.
Figure 5.24 Simulation of the Bioluminescence system using the Matlab ode113 solver with the input $A_{\text{source}} = 0$

Figure 5.25 Simulation of the Bioluminescence system using the Matlab ode113 solver with input $A_{\text{source}} = 10$
Figure 5.26 Simulation results of the Bioluminescence system using the ode15s solver with the input $A_{\text{source}} = 0$. Reprinted from [9].

Figure 5.27 Simulation results of the Bioluminescence system using the ode15s solver with the input $A_{\text{source}} = 10$. Reprinted from [9].
5.5 Stochastic simulation of the Bioluminescence system

In the previous section, it has been observed that the bioluminescence system is modeled using the ordinary differential equations (ODE) approach and this ODE approach required the various processes to be converted into chemical equations first and then later on converted to differential equations. However, the ODE approach has an obvious limitation which indicates that it does not account for spatial aspects. The time-evolution of a system is considered to be a more continuous and predictable approach by the mathematical or the deterministic method and this time-evolution of the system is actually governed by a set of coupled ordinary differential equations. These set of coupled ordinary differential equation is better known as the “reaction-rate equations”. The time-evolution of the system is treated as some sort of random process by the stochastic approach and is governed by a single differential equation which is known as the “Stochastic Master Equation”. The advantage of the stochastic simulation methods is that it allows one to study the distribution of the population of the species at a particular time [32].

In our research, we developed the stochastic model to verify the robustness of the bioluminescence model in the presence of some noise and also the random behavior of the biochemical reactions. The stochastic model described in the research is based on the Next Reaction Method. The Next Reaction Method proposed by Gibson and Bruck [14] is an enhancement to Gillespie’s original algorithms namely the Direct Method and the First Reaction Method. Although Gillespie’s Direct Method and the First Reaction Method produce the same results, they differ strongly in their performance. The difference in the performance is mainly attributed to the use of different operations. The Direct Method uses only two random numbers per iteration whereas the First Reaction Method uses (M+1) random numbers per iteration where, ‘M’ represents the number of reactions.
Gibson and Bruck’s Next Reaction Method as already mentioned is an improvement over Gillespie’s Algorithm. The re-calculation of all the reaction propensities is avoided in this method. A data structure named the dependency graph identifies the reactions whose propensities are to be updated. Another data structure concept named the priority queue is implemented to store the putative time values calculated based on the propensity functions, thereby avoiding the recalculation of all the putative times for each iteration. In our research, the stochastic simulation of the Bioluminescence system is carried out using the Next Reaction Method.

The Bioluminescence model is described by 24 biochemical reactions. The volume of the system is fixed at 10e-03 liters. The simulation was carried out for a period of 1500 seconds to match with the simulation of the ODE model described in the previous section. The stochastic model was simulated by using the parameters described in the Table 5.4. The stochastic simulation waveform is shown in Figure 5.29. From the figure, it has been observed that the stochastic model confirms the production of LuxI and LuxA/B when an auto inducer is introduced into the system.

<table>
<thead>
<tr>
<th>Species</th>
<th>A_i</th>
<th>LuxR</th>
<th>LrA_i</th>
<th>LrA_i2</th>
<th>LuxD</th>
<th>LuxD_complex</th>
<th>ipromR</th>
<th>DNA_loop</th>
<th>mRNAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Molecules</td>
<td>10</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Species</th>
<th>mRNAI</th>
<th>LuxI</th>
<th>A_source</th>
<th>Luxbox</th>
<th>promI</th>
<th>promR</th>
<th>ipromI</th>
<th>LuxA/B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Molecules</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 5.6 Initial values (number of molecules) of the biochemical species for the stochastic model of the Bioluminescence system
Figure 5.28 Stochastic simulation of the bioluminescence system using the Next Reaction method showing the input $A_i$ and output LuxA/B. LuxR and LuxI are also included.

Figure 5.29 Stochastic simulation of the bioluminescence system using the Gillespie Algorithm showing the input $A_i$ and output LuxA/B. LuxR and LuxI are also included [9].
Figure 5.30 Figure showing the luminescence response of a cell in a colony [5].

From Figure 5.29 it is observed that the results from the stochastic simulation of the Bioluminescence system using Gibson and Bruck’s Next Reaction Method are similar to those obtained by R. Krishnan [9] whose results are plotted in Figure 5.30. The concentration of the auto inducer increases steadily with time in the stochastic simulation carried out by R. Krishnan [9] in his research. The same result is obtained with the stochastic simulation that is carried out using Gibson and Bruck’s Next Reaction Method. Further, the simulation time is less for simulating the bioluminescence system using the Next Reaction Method when compared to the Gillespie Algorithm. In the ABM, the concentration of the auto inducer increases steadily and after a certain time interval it remains constant unlike the stochastic simulation method where the concentration of the auto inducer increases steadily throughout the simulation time. The two models (Stochastic and ABM) cannot be quantitatively compared because there are differences in the way the molecular species are modeled in both cases. For example the promoters are modeled as concentrations in the stochastic model but they are modeled as three dimensional
structures in the ABM, the same applies to modeling of the operator regions. Also relative reaction rates are used in the ABM. Nevertheless both the models agree very well with each other at a qualitative level. The major drawback of the ABM is that it is a bottom-up approach and modeling a system with the bottom-up approach requires that every individual agent’s behavior to be described in detail. The more details we require about the behavior of each individual agent, the greater the computational power that is required simulating the behavior of all these agents.
6 Conclusions and Future Work

6.1 Introduction

In this chapter, a brief summary of our research is given and also a few conclusions of our study are stated. Suggestions are also provided for future work.

6.2 Conclusions

In our thesis, we have described clearly the need to model biological pathways and also the necessity to perform certain simulated mutations on them. We have also made a brief mention about the occurrence of stochasticity in the biological systems and with this we have discussed the need for stochastic methods for simulating biological pathways and their advantages when compared to experimental methods for carrying out the biological experiments. In our research, we have described the different stochastic methods that exist and made a comparison of the various stochastic methods described. Also explained in this thesis are the various advantages of the stochastic simulation methods, their limitations when compared to the Agent Based Modeling method and also the advantages and drawbacks of the Agent Based Modeling method.

We have carried out the stochastic simulation of a biological pathway, namely the Phage Lambda system, by using Gibson and Bruck’s Next Reaction Method and compared the results of this simulation with those of R. Krishnan [9]. We have also compared our results with those of V. Vallurupalli [13] whose work involved modeling the Phage Lambda system using an Agent Based Modeling technique. In our thesis, we have modeled the Phage Lambda system and the Bioluminescence system using Gibson and Bruck’s Next Reaction Method. The Phage Lambda system exhibits the switching behavior which is quite similar to an electrical inverter. Due to
this mechanism, the Phage Lambda system is considered to be a bio-inverter and thus finds a very significant application in developing circuits using bio-components [6]. Experiments that are carried out in the laboratory or in vivo are always time consuming and expensive and hence are not ideal solutions. Thus, the stochastic simulation enables the modeling of the bio-chemical reactions that are important for developing bio-circuits.

The bioluminescence in the V. Fischeri bacteria is obtained by a process called quorum sensing. Based on the population density of the V. Fischeri bacterium, the quorum sensing regulates and coordinates the gene expression. The V. Fischeri bacterium releases a chemical called auto-inducer and the quorum sensing mechanism is regulated by this auto-inducer. The stochastic model enables the modeling of the V. Fischeri bacterium.

### 6.3 Suggestions for future work

There is plenty of scope to expand the research in the area of simulation of biological pathways. The areas where there is scope for future work are summarized below:

1) **Application of the Next Reaction Method to the Bioluminescence system and the TNF α-mediated NF-κβ pathway:** The Next Reaction Method is an improvement over the Gillespie Algorithm which improves the performance with respect to the time while still maintaining the exactness of the algorithm. The Gillespie Algorithm normally requires large computational time for a system that has a large number of reactions. This is overcome in the Next Reaction Method proposed by Gibson and Bruck [14]. The current research dealt with the application of the Next Reaction Method to the Phage Lambda system and the same can be extended to other biological pathways such as the Bioluminescence System and the TNF α-mediated NF-κβ pathway [9].
2) **Parallel algorithms:** Though the stochastic simulation method accounts for the spatial properties while simulating the biological pathways, it has its own drawbacks. The stochastic simulation method requires computational power and hence it takes a really long time to simulate a realistic biological system. One of the several techniques that have been developed to overcome this problem is the parallelizing of the algorithms [4]. One parallel algorithm approach would be to use grid technology. Grid technology is the process by which computer resources from several different domains are combined and applied to a common task which requires a large number of computer processing cycles or the necessity to process huge amounts of data. Grid computing is a kind of network-distributed parallel processing [69].

3) **Spatio-temporal algorithms:** In the biological world, there are many components which interact in a three dimensional space. Hence, the stochastic spatio-temporal simulation of the biological system is required. The spatio-temporal algorithms are easily handled by enhancement in the performance of the Gillespie Algorithms. A. B. Stundzia and C. J. Lumsden [70] and Elf *et al.* [71] have extended the Gillespie algorithms so that it models intracellular diffusion too. T. S. Shimizu [37] has extended the Stochsim algorithm so as to include the spatial effects of the system. Another effort in this direction is MCell [72]. These methods can be applied to the systems considered here.

4) **Other stochastic approaches:** Other approaches such as the optimized direct method (ODM) [12], the sorting direct method (SDM) [12], or the logarithmic direct method [17], can also be tried.
References:


[15] Deterministic model-biology online:


[45] Dizzy: Chemical Kinetics Stochastic Simulation Software:  


[47] Stochastic simulation using MATLAB:  


