MONTE CARLO SIMULATIONS OF BIOMOLECULAR REACTION NETWORKS MODELED ON PHYSICAL PRINCIPLES

by

Isaac Tian Shi Li

A thesis submitted in conformity with the requirements

for the degree of Master of Applied Science

Graduate Department of Biomaterials and Biomedical Engineering

University of Toronto

Abstract

Monte Carlo Simulations of Biomolecular Reaction Networks

Modeled on Physical Principles

Isaac Tian Shi Li

Master of Applied Science

Graduate Department of Biomaterials and Biomedical Engineering

University of Toronto

2007

Understanding and engineering complex biomolecular networks in the cell are the goals of systems and synthetic biology. The effects of localization, spatial heterogeneity and molecular noise in biomolecular networks are not well understood. In this research, a theoretical approach to accurately simulate large biomolecular networks using the Monte Carlo method was introduced. Incorporating this theory, a computational tool named Monte Carlo Biomolecular Simulator (MBS) was developed, enabling studies of biomolecular kinetics with both spatial and temporal resolutions. The accuracy of MBS was verified by comparison against the classical deterministic approach. Furthermore, the effects of localization, spatial heterogeneity and molecular noise were studied in three simulated systems, showing their huge impact on the overall reaction kinetics. Lastly, the MBS was used as an engineering tool to create and fine-tune a synthetic protein network analogous to a D-latch memory unit commonly used in electrical circuits.

Acknowledgement

I would like to thank Prof. Kevin Truong for his supervision. His patience, encouragement and guidance helped me tremendously to find new light in research even in the most hopeless situations. I want to thank Elizabeth Pham for her help and support in the lab. I also want to thank Ranjith K.R., Amir Manbachi, Warren Shum and Stanley Wong for their collaborations with me. It was a fun and inspiring experience working with everybody from this lab.

I would like to thank my parents for their cheers, understanding and selfless support through my ups and downs. I want to thank Rachel for always being there to share my joy and pain as well as her understanding and encouragement. And last but not least, I want to thank all the friends around me for two years of invaluable experience in graduate school.

Table of Contents

List of Tables ..... **viii**

List of Figures ..... **ix**

List of Appendices ..... **xvi**

List of Publications ..... **xvii**

Chapter 1 Introduction **1**

1.1 Motivation and background 2

1.2 Key objectives 4

1.3 Organization 5

Chapter 2 Literature research **6**

2.1 Introduction 7

2.2 Circuit engineering analogy for systems and synthetic biology 8

2.3 Computation in synthetic biology 10

2.3.1 Elucidating natural biological network kinetics 10

2.3.2 Synthetic networks constructed 12

2.3.3 Applications of synthetic biology 15

2.4 Quantitative computational modeling approaches 16

2.4.1 Deterministic chemical kinetics approach 19

2.4.2 Stochastic kinetics approach 20

2.5 Summary 22

Chapter 3 Theory development 24

3.1 Introduction 25

3.2 Random walk model of diffusion 26

3.3 Simulation of random walk 31

3.3.1 Random number generation with non-uniform distribution 31

3.3.2 Random 3D-isotropic sampling: 33

3.3.3 Diffusion step size 35

3.4 Molecular collisions from random walk 38

3.5 Association reactions from a microscopic perspective 45

3.6 Association reaction - relating microscopic and macroscopic behaviour 50

3.7 Dissociation reactions - relating microscopic and macroscopic behaviour 54

3.8 Summary 55

Chapter 4 MBS: Monte Carlo Biochemical Reaction Simulator 57

4.1 Introduction 58

4.2 Implementation 58

4.2.1 Program architecture 59

4.2.2 Data structure 61

4.2.3 Handling diffusion in restricted regions 65

4.2.4 Handling single reactions 66

4.2.5 Handling multiple reactions 67
4.2.6 Random number generation 68
4.2.7 Optimizations 68
4.3 Using MBS 71
4.3.1 Running MBS 71
4.3.2 Viewing molecule movie 73
4.3.3 Input scripts 75
4.3.4 Output file format 79
Chapter 5 Model systems 81
5.1 Overview 82
5.2 Diffusion verification 83
5.2.1 Method 83
5.2.2 Results and discussions 83
5.3 Basic reaction kinetics verification 86
5.3.1 Method 86
5.3.2 Results and discussions 87
5.4 The predator-prey model 89
5.4.1 Method 89
5.4.2 Results and discussions 90
5.5 Genetic oscillator 92
5.5.1 Method 92
5.5.2 Results and discussions 93

5.6 Ca2+ wave 95

5.6.1 Method 95

5.6.2 Results and discussions 96

5.7 A protein reaction network that implements a D-latch memory element 100

5.7.1 Method 100

5.7.2 Results and discussions 100

Chapter 6 Conclusion 107

6.1 Ongoing projects and future work 109

References 110

List of Tables

Table 2-1: Software tools developed for the modeling and simulation of biological interactions 17

Table 2-2: Databases developed to store, categorize, and share data from biological studies and modeling 18

Table 4-1: Region types and definition parameters. 65

Table 4-2: Details included in the setup scripts for each simulation 75

List of Figures

Figure 2-1: Examples of synthetic genetic and protein circuits. A. a negative-feedback circuit consisting of a single gene: the promoter drives the transcription of the target gene and a gene coding for a repressor, the expression of which regulates its own promoter through inhibition. B. a toggle switch: promoter 1 drives the expression of repressor 2, which inhibits the transcription of the target gene by blocking promoter 2; this second promoter drives the expression of the target gene and repressor 1, which inhibits the transcription of promoter 1, in effect activating the transcription of the target gene; in the presence of inducers, inhibition effects can be blocked, further regulating the expression of the target gene. C. an oscillator circuit: promoter 1 drives the expression of repressor 3, which blocks promoter 3; promoter 3 drives the expression of repressor 2, inhibiting the promoter 2 responsible for driving the expression of repressor1 to block the transcription of promoter 1. D. an AND gate: the protein (white) is active only in the presence of both input proteins. E. an OR gate: the protein is active when one or both input proteins are present. F. a NOT gate: the protein is no longer active in the presence of an input protein. 13

Figure 3-1: The diffusion probability density function. The x-axis is the radial distance from the origin of diffusion. Plotted with parameter: D=10^{-10}m^2/s, t=100ns. The graph assumes the shapes of a Gaussian function. From this graph, the diffusion can be estimated to be in the order of 10nm, roughly 1% of the size of a cell. 29

Figure 3-2: The time-evolution of the diffusion probability density function. The x-axis is the radial distance from the origin of diffusion. Plotted with parameter D=10^{-10}m^2/s, the time ranges from t=0.5s to t=5 s at a step of 0.5 s..... 30

Figure 3-3: Probability distribution function of arbitrary shape integrated from x_1 to x. The x-axis is the desired random number, the y-axis is the probability distribution of the random number x. 33

Figure 3-4: Spherical coordinate system, r, θ, and φ are defined as shown. 33

Figure 3-5: Comparison of: A. uniform random sampling of θ and ϕ and B. corrected random sampling of θ and ϕ. The uniform sample of θ and ϕ causes higher density at the polar regions and low density at the equator, hence, the distribution is not isotropic. 34

Figure 3-6: Comparison of the density distribution and population distribution in 2D. The numbers shown in each ring indicate the unit density / population within that ring. The population within each ring is then the area of that ring multiplied by the density in that ring. If the density distribution peaks at the centre as in a Gaussian distribution, the population would peak at some distance away from the centre. 36

Figure 3-7: Plot of *R(r)*. Horizontal axis is the radial distance $\frac{r}{\sqrt{4Dt}}$ in spherical coordinate, the vertical axis is R, indicating that if the variable R were to be random sampled, the radial distribution will achieve Gaussian density distribution. 38

Figure 3-8: Illustration of random walk collision. The wiggly lines are the actual path each molecule takes to get to the collision site. The arrows indicate the vector from

molecule original location to the collision site, which gives the probability they will land there. The box is the volume in which collision would occur. 40

Figure 3-9: The time evolution of collision probability density in 2D. The horizontal plane indicates the physical space where molecules move and collide. The vertical axis is the collision probability density. The parameters used here are: D_A=D_B =10^{-10} m^2/s, Protein diameter=5 nm, V_c=5.2e-25 m^3, l_0=10 nm, time range=[10ns, 100ns] at 10ns interval. The peak is located at the centre of the two reagent molecules. 41

Figure 3-10: Spatial variation of the collision probability density function. The horizontal axis is time, the vertical axis is the collision probability function. Each curve shows the cPDF at a distance from the maximum location of the cPDF function (from 10 nm to 20 nm at 2 nm intervals). Parameters used in this plotting are: D_A=D_B =10^{-10} m^2/s, Protein diameter=5 nm, V_c=5.2e-25 m^3, l_0=10 nm. 42

Figure 3-11: Collision probability function with respect to time. The following parameters are used: D_A=D_B =10^{-10} m^2/s, protein diameter=5 nm, V_c=5.2e-25 m^3, l_0=10 nm. 44

Figure 3-12: The accumulative collision probability function $\int_0^t P_c(t',l_0)dt'$ as a function of time. 46

Figure 3-13: Probability of reaction as a function of the duration of collision state $<t_c>$. The horizontal axis is in unit of τ_r. 48

Figure 4-1: Software architecture. Calling between classes, files and data structures. Solid black: program internal calling between classes. Solid grey: file I/O. Dotted black: memory data structure access..... 61

Figure 4-2: Running MBS screenshot. Initialization and parsing setup, experiment and geometry scripts. 72

Figure 4-3: Running MBS screenshot. Simulation and total runtime. 73

Figure 4-4: MBSmovie displaying a reaction in a spherical region. The fps and current step index are displayed on the left upper corner. The colour map for different molecules is displayed on the upper right corner. On-screen instructions are displayed at the bottom of the screen. 75

Figure 4-5: Format of the setup script. 77

Figure 4-6: Format of the experiment script. 78

Figure 4-7: Format of the geometry script. 78

Figure 4-8: Format of kinetics data output file. 79

Figure 4-9: Format of reaction movie output file. 80

Figure 5-1: A. The population distribution of 10^4 molecules was diffused from a single point as a function of their distances from the origin of diffusion. The three distinctive humps are the population distribution of different total time durations of 10 ms, 100 ms and 1000 ms. The three shades are simulations under different time step duration Δt=10^{-3} s (lighter grey), 10^{-4} s (darker grey) and 10^{-5} s (black). The coincidence of the three shaded curves shows that diffusion is independent to the Δt. The same diffusion coefficient D=10^{-10} m^2/s is used. The population distribution of B. uniform step size distribution model and C. our diffusion model show good agreement with deterministic solution for our model and disagreement for the uniform distribution model. Both simulations have a total duration of 0.1 s. The deterministic solution is indicated as the thick grey line. The population

distributions using 5 time step durations Δt=10^{-2} s (solid square), 5×10^{-3} s (hollow square), 10^{-3} s (solid triangle), 10^{-4} s (hollow triangle), and 10^{-5} s (solid circle) were compared with the deterministic solution. 85

Figure 5-2: A. Population of reagent molecule as a function of time plotted in log scale for association reactions with various rate constant k_a ranging from 10^3 to 10^8 $M^{-1}s^{-1}$. B. Population of reagent molecule as a function of time plotted in log scale for dissociation reaction with various rate constant k_d ranging from 10^{-1} to 10^3 s^{-1}. C. Reversible reaction kinetics showing population of reagent molecule as a function of time plotted in log scale for association reactions with various simulation time step durations Δt=10^{-6} s to 10^{-3}s. In all the above figures, the simulated kinetics curve (thin black) is compared to the deterministic kinetics curve (thick grey). The Δt is 0.0001s and the total duration is 0.5 second. D. equilibrium constants in k_a-k_d space matches what is predicted by the deterministic kinetics equations. 88

Figure 5-3: Deterministic model solutions of A. phase space predator population vs. prey population, B. predator (grey) and prey (black) population over time, C. zoomed in view of the kinetics curves. Monte Carlo simulation solution of D. phase space predator population vs. prey population, E. predator (grey) and prey (black) population over time, F. zoomed in view of the kinetics curves. G. spatial heterogeneity of the reaction, showing large spatial fluctuation. Black dots: prey, grey dots: predators. F. demonstration of diffusion coefficient changes the frequency and amplitude of the simulation in phase space. Lower diffusion coefficient (lighter grey) results in curves with greater amplitudes while higher

diffusion coefficient (darker grey) shifts the curves towards the bottom left corner. The deterministic solution is shown in black..... 91

Figure 5-4: A, the plasmid schematic of a genetic oscillator. Regions with arrows were repressor binding regions of the plasmid DNA. Regions marked with λcI, TetR and LacI are the genes regulated by their respective repressor binding regions. B, the population changes of the three repressor proteins in the prokaryotic model. C, D, the population oscillations of the three repressor proteins in the eukaryotic model with C, low nuclear membrane transport rates of 5×10^{3} s^{-1} and D, high nuclear membrane transport rate of 5×10^{5} s^{-1}. Dotted lines represented λcI; solid black, TetR; solid grey, LacI..... 95

Figure 5-5: The spatial propagation of the Ca^{2+} ions in time. Top-left corner shows the triggering event at t=0 ms. The ion propagation to the right can be seen moving much faster than diffusion, hence the ion distribution profile is elongated in the horizontal direction. 97

Figure 5-6: Graphical representation of the Ca^{2+} distribution after 10 ms for experiments with different channel binding/opening rate: A, 10^{8} $M^{-1}s^{-1}$, B, 10^{7} $M^{-1}s^{-1}$, and C, 10^{6} $M^{-1}s^{-1}$. D, the Ca2+ population outside the compartment over time for 1000 membrane channels. E, Ca2+ release rate over time for 1000 membrane channels F, the Ca2+ population outside the compartment over time for 100 membrane channels. G, Ca2+ release rate over time for 100 membrane channels. Black, dark grey and light grey lines in each figure (D, E, F, G) are three simulations using the same setup parameters. 99

Figure 5-7: A. A clock inverted D-latch in electronics representation, the four NOR gates are names N_{data}, N_{clock}, $N_{output1}$, and $N_{output2}$. B. Protein circuit implementation of the D-latch. C. the timing diagram of electronic D-latch in comparison with protein D-latch. Grey lines indicate the response expected for digital circuit, black lines are the response from the protein circuit..... 105

List of Appendices

Appendix A 120

Appendix B 125

Appendix C 135

List of Publications

Journal articles

[1] I. T. S. Li and K. Truong, "A computation tool for Monte Carlo simulations of biochemical reactions modeled on physical principles," *Bioinformatics (submitted)*, 2007.

[2] I. T. S. Li, W. Shum, and K. Truong, "160-fold acceleration of the Smith-Waterman algorithm using a field programmable gate array (FPGA)," *BMC Bioinformatics*, vol. 8, pp. 185, 2007.

[3] I. T. S. Li, K. R. Ranjith, and K. Truong, "Sequence reversed peptide from CaMKK binds to calmodulin in reversible Ca2+ -dependent manner," *Biochem Biophys Res Commun*, vol. 352, pp. 932-5, 2007.

[4] I. T. S. Li, E. Pham, and K. Truong, "Protein biosensors based on the principle of fluorescence resonance energy transfer for monitoring cellular dynamics," *Biotechnol Lett*, vol. 28, pp. 1971-82, 2006.

[5] I. T. S. Li and K. Truong, "FRET evidence that an isoform of Caspase-7 binds but does not cleave its substrate," *Biochemical and Biophysical Research Communications (submitted)*, 2007.

[6] E. Pham, J. Chiang, I. T. S. Li, W. Shum, and K. Truong, "A computational tool for designing FRET protein biosensors by rigid-body sampling of their conformational space," *Structure*, vol. 15, pp. 515-23, 2007.

[7] E. Pham, I. T. S. Li, and K. Truong, "Computational Modeling Approaches for Studying of Synthetic Biological Networks," *Current Bioinformatics (submitted)*, 2007.

[8] J. J. Chiang, I. T. S. Li, and K. Truong, "Creation of circularly permutated yellow fluorescent proteins using fluorescence screening and a tandem fusion template," *Biotechnol Lett*, vol. 28, pp. 471-5, 2006.

Book chapter

[9] I. T. S. Li, E. Pham, and K. Truong, *Current approaches for engineering proteins with diverse biological properties*: Landes Bioscience, 2007. (To be published in Oct. 2007)

Conference proceedings

[10] I. T. S. Li and K. Truong, "Monte Carlo simulation of biological reactions with spatial and temporal resolution," presented at 2nd Annual Canadian Student Conference in Biomedical Computing, London, Ontario, Canada, 2007.

[11] I. T. S. Li, W. Shum, and K. Truong, "160-fold acceleration of Smith-Waterman algorithm using field programmable gate array (FPGA)," presented at 2nd Annual Canadian Student Conference in Biomedical Computing, London, Ontario, Canada, 2007.

[12] I. T. S. Li, J. J. Chiang, and K. Truong, "FRET evidence that an isoform of caspase-7 binds but does not cleave its substrate," presented at 28th Annual International IEEE Engineering Conference in Medicine and Biology New York, 2006.

[13] J. J. Chiang, I. T. S. Li, and K. Truong, "The FPMOD: A modeling tool for sampling the conformational space of fusion proteins," presented at 28th Annual International IEEE Engineering Conference in Medicine and Biology, New York, 2006.

Chapter 1

Introduction

1.1 Motivation and background

Cells are complex biological systems made of networks of biomolecular reactions. The study of the kinetic behaviour of these networks is crucial to understanding the intricacies of their resulting cellular behaviours, which is the goal of the emerging new field of systems biology. Most biomolecular studies to date have focused on the interactions and reaction mechanisms among a small number of proteins and other biomolecules such as nucleotides, metabolites and ions. This pioneering work has provided the foundation to study larger and more sophisticated biomolecular reaction networks in the cell. These networks have distinct characteristics. First, biomolecules in the cell such as proteins, DNA, ions and metabolites are spatially organized into compartments or anchored to membranes. It is still largely unknown how the localization of these molecules provides signaling, control and functional advantages to the cell. Second, many biomolecular reactions such as the Ca^{2+} wave and the action potential involve both temporal and spatial kinetics. Thus, a molecular concentration profile in time and space is critical for studying these networks. Lastly, restricted by the size and organization of the cell, the population of certain molecular species such as protein coding DNA could be as low as single digits. Reactions involving low copy number species have large statistical fluctuation in activities, the behavior of which is important to the stability and variability to the protein network. Again, it is still largely unknown how this impacts cellular behaviour. Therefore, being able to simulate all the above aspects of biomolecular reactions is crucial for understanding their resulting cellular behaviours.

Beyond understanding, the ultimate goal is to be able to modify and control biological systems. This is where synthetic biology emerges, which strives to build increasingly complex biological networks through the integration of molecular biology and engineering. The growth of the field has been supported by progress in the design and construction of synthetic genetic and protein networks. This has led to the possibility of assembling modular components to attain novel biological functions and tools. In addition, these synthetic networks give rise to insights that facilitate the investigation of interactions and phenomena in naturally-occurring networks. Integration of well-characterized biological components into higher order networks requires computational modeling approaches to rationally construct systems that are directed towards a desired outcome. A computational approach would improve the predictability of the underlying mechanisms that would otherwise be difficult to deduce through experimentation alone. The analysis and interpretation of quantitative models becomes increasingly important towards taking a systems-level perspective on synthetic genetic and protein networks.

There are in general two classes of approaches to simulate biomolecular reactions: deterministic and stochastic. The most common deterministic approach to study biomolecular kinetics is by ordinary differential equations (ODE), which usually assumes homogeneous concentration for all biomolecular species within the reaction volume. Therefore, ODE would not provide the desired spatial resolution. One could obtain spatial information by applying a system of second order partial differential equations (PDE). However, deriving and solving a large system of equations needed for spatial resolution becomes more challenging as the number of interactions increases, often requiring many

approximations. Furthermore, the effect of molecular fluctuation for low copy number molecular species cannot be easily handled by a deterministic model. To resolve these problems, we created a computational tool named MBS (Monte Carlo Biomolecular Reaction Simulator) that used a stochastic model to simulate the motion and reaction of each molecule in user-defined spaces. Although a few similar attempts have been made recently to simulate signaling pathways and biomolecular oscillations, the models used in these studies adopted convenient but crude approximations of physical principles such as the handling of single molecule diffusion and reaction probability. Thus, using the same physical approximations, we could not produce simple diffusion and reaction constants. Consequently, in our computational tool (MBS), we developed a set of physically realistic models to represent the molecular diffusion and reaction processes.

1.2 Key objectives

The key objectives of this research are:

- Develop a theory to accurately represent molecular diffusion and chemical reactions using Monte Carlo method.
- Develop a software package incorporating the theory for studying biomolecular networks that provides both temporal and spatial kinetics information.
- Verify the theory and test the software by comparing simulation results with deterministic results under known conditions.
- Demonstrate the important effects of molecular localization, spatial heterogeneity and molecular fluctuation to overall chemical kinetics.
- Design a synthetic protein network behaving like a digital circuit.

1.3 Organization

The remaining chapters of this thesis are organized as follows:

- **Chapter 2** provides a literature survey on biological systems modeling, showing what has been done and why this research is needed.
- **Chapter 3** introduces the theory and method to describe diffusion and chemical kinetics in Monte Carlo simulations, forming the basis for the software development.
- **Chapter 4** provides details on the architecture and implementation of the MBS software, where the technical difficulties and solutions are addressed.
- **Chapter 5** presents simulations using MBS including the verification of its validity and demonstration. Models showing the effect of spatial heterogeneity, molecular localization and fluctuation were constructed. A chemical memory unit was designed using protein network that behave like a latch in electric circuit. The design was realistic and could be created using protein engineering.
- **Chapter 6** summaries the thesis and discusses the future direction of this research.

Chapter 2

Literature research

The content of this chapter was modified from the peer-review journal paper:

E. Pham, I. T. S. Li, and K. Truong, "Computational Modeling Approaches for Studying of Synthetic Biological Networks," *Current Bioinformatics (submitted)*, 2007.

2.1 Introduction

Systems biology aims at understanding systems as a whole by studying the interactions between the components of biological networks. One goal of systems biology is to provide an in-depth comprehensive body of knowledge of the interactions and kinetics governing biological systems at the molecular level. Synthetic biology encompasses an engineering-based approach to designing biological networks. It shares the holistic perspective of systems biology, as its ultimate goal is to construct *de novo* networks of high complexity and interconnectivity. Progress in synthetic biology will address fundamental principles of biological interactions, as well as lead to practical applications in drug discovery and biotechnology. In order to move towards a higher-order, systems-level perspective, it is necessary to examine the composition, structure, and kinetics of cellular networks, rather than the characteristics of the isolated parts alone.

An important post-genomic research area is the analysis and elucidation of the dynamic interactions of genes and proteins in naturally occurring systems. Linkages between the molecular and system levels were recently made possible by advances in functional genomics and proteomics. The current drive is to analyze systems in terms of their responses to perturbations and to uncover network features such as robustness and degeneracy.

Molecular characteristics of biological interactions have been identified and categorized into specific functional modules. A systematic means of piecing together different modules to progressively build more complex networks will not only lead to a systems-level

understanding, but also reveal the underlying kinetics governing how the individual modules interact and respond to each other. This results in a more continuous stream of knowledge, from molecular to systems descriptions. Not only is there a gap in our understanding and knowledge of all biological phenomena, even for biological systems in which all the components are known, it is still unclear precisely how these components interact to make cellular processes work. The vast amount of biological data from molecular biology has revealed many sequences and properties of genes and proteins, but is not sufficient for interpreting system behaviour.

Recently, computational modeling approaches have been employed to study natural biological systems and would be applicable, in fact highly recommended, for synthetic networks. These approaches integrate advances in algorithms and statistics to analyze biological data. Through a combination of both experimental and computational approaches, we can gain deeper understanding of the function of biological processes. Therefore, it is worthwhile to look at how computational approaches could be used to complement construction and experimentation of synthetic networks.

2.2 Circuit engineering analogy for systems and synthetic biology

Biological networks are analogous to electrical circuits. Both circuits and biological systems transform information from one form into another based on a set of defined rules. Stimuli function as inputs, while signals modulating the behaviour of the system are processed and generated as outputs.

Mapping out the components of a biological system and making the connections between interacting components is analogous to drawing a circuit diagram. In order to deduce the mechanisms controlling these biological circuits, a parts list needs to be generated and the transformation between input and output needs to be established. While the former has been successful through genomic sequencing and protein studies, the latter requires more rigorous analysis. Building a circuit to perform a particular function is much easier than deducing the function of an existing black-box circuit solely through correlating its outputs with its inputs. Circuit control theory has been used to develop a theoretical understanding of an adaptation mechanism through negative feedback [14, 15]. However, this approach has limitations as control theory assumes that inputs are provided to the system, but in biology, such inputs or stimuli are often created and refined continuously within the system itself. In another study, an integrative modeling approach was used to run a circuit simulation of the lysis-lysogeny decision circuit of bacteriophage lambda, making use of the parallels between genetic and electrical circuits [16]. Similarly, other frameworks integrating control theory and biological control processes have been proposed to describe genetic regulatory networks and adaptation in bacterial chemotaxis [15, 17]. These function well as system descriptions, drawing parallels between biological processes with more established control theory. While such an analogy allows for a framework in which to systematically identify and analyze synthetic biological networks, it does not address the need to computationally study these networks for a more quantitative perspective. However, even for well-studied systems, no set of defined equations or approximations correspond exactly to how that system behaves. In many cases, even the smaller components making up a biological system are still under study. Hence,

there is still much characterization on the biological level to be done before a systematic characterization and understanding of the real biological networks can be understood.

2.3 Computation in synthetic biology

Although simulations at the systems biology level is currently limited, synthetic biology may lend insights to explain phenomena observed in real biological systems. The rational construction and analysis of synthetic networks provides a framework for computational modeling studies. The construction of useful and predictive synthetic networks allows the direct prediction and measurement of model parameters. As both the complexity and design of the networks are under control of the designer, there are fewer ambiguities and uncertainties. As well, there is a firmer foundation upon which more complex networks can be built.

2.3.1 Elucidating natural biological network kinetics

The design of synthetic networks allows chosen subnetworks of natural biological systems to be isolated. Modeling and experimental studies can be focused first on understanding the isolated subsystem before progressively increasing complexity. Accurate models of synthetic networks provide fundamental insights and act as a foundation with which to describe natural biological networks, including genetic regulatory networks and protein signalling pathways. The ultimate goal of synthetic biology is to construct increasingly complex networks, concomitantly assembling increasingly more complete models of natural systems. The advantage of this approach is that at each stage, subsystems have been

characterized by modeling and experimentation, thus keeping the number of unknowns at a minimum. Practically, this approach reduces the degree of trial-and-error experimentation required for the understanding of complex biological networks. Once the structures of synthetic networks are mapped out and their functional dynamic properties are understood, an ever-growing library of circuits will facilitate the classification and comparison of subsequent circuits to provide yet more insights into the complexity of natural biological systems. Synthetic biology allows the study of natural regulatory networks and cellular behaviours using *de novo* networks, potentially leading to future applications in biotechnology and medicine.

Undoubtedly, signalling networks are complex and highly interconnected, interacting at several levels to regulate biological functions within cells [18]. Synthetic genetic regulatory systems mimicking those of mammalian cells have led to the potential of designing mammalian cells with desired properties for tissue engineering, gene therapy, and biopharmaceutics [19]. Furthermore, many diseases result from malfunctioning of natural biological networks including both signalling pathways and transcriptional regulation. In diseases like cancer, single abnormalities in signalling pathways do not lead to complications, but the combined effect of multiple abnormalities to several key pathways result in substantial consequences. Understanding how individual components function within the context of a larger, complex signalling network provides a molecular view of which interactions are involved in causing the diseased state.

2.3.2 Synthetic networks constructed

Using the analogy of logic flow from circuit engineering, both genetic-based and protein-based synthetic networks have been designed and tested [20]. A library of networks with novel connectivities between transcriptional regulators and the corresponding promoters was previously developed for combinatorial synthesis of biological networks of varying levels of complexity [20]. Examples of synthetically engineered gene circuits include autoregulatory systems displaying stability through negative feedback, toggle switches, logic gates, and repressilators [21-23]. (Figure 2-1A, B, C) Similarly, engineered protein circuits have been constructed to function as Boolean logic gates of AND, OR, and NOT [24, 25]. (Figure 2-1D, E, F)

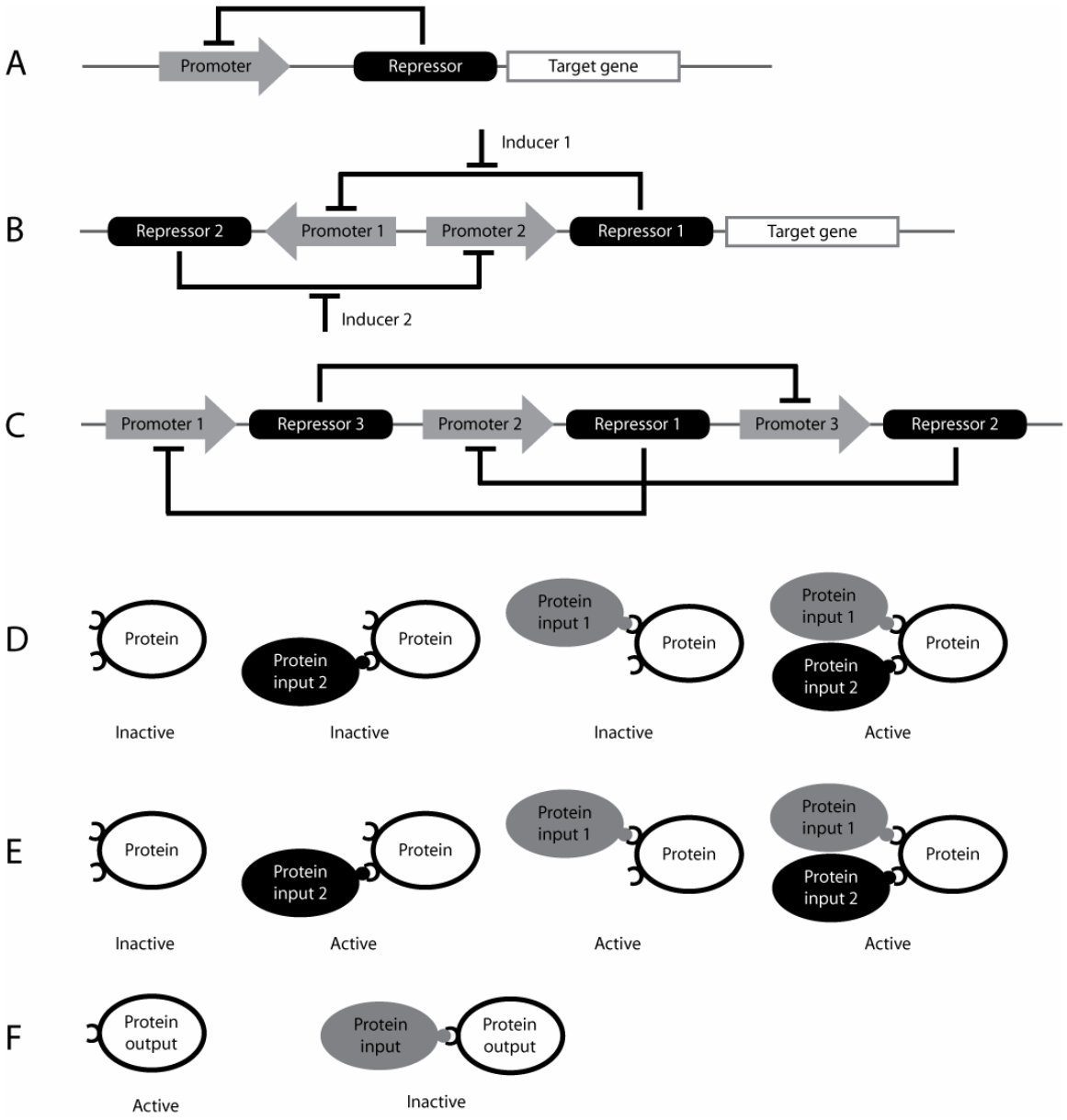

Figure 2-1: Examples of synthetic genetic and protein circuits. A. a negative-feedback circuit consisting of a single gene: the promoter drives the transcription of the target gene and a gene coding for a repressor, the expression of which regulates its own promoter through inhibition. B. a toggle switch: promoter 1 drives the expression of repressor 2, which inhibits the transcription of the target gene by blocking promoter 2; this second promoter drives the expression of the target gene and repressor 1, which inhibits the transcription of promoter 1, in effect activating the transcription of the target gene; in the presence of inducers, inhibition effects can be blocked, further regulating the expression of the target gene. C. an oscillator circuit: promoter 1 drives the expression of repressor 3, which blocks promoter 3; promoter 3 drives the expression of repressor 2, inhibiting the promoter 2 responsible for driving the expression of repressor1 to block the transcription of promoter 1. D. an AND gate: the protein (white) is

active only in the presence of both input proteins. E. an OR gate: the protein is active when one or both input proteins are present. F. a NOT gate: the protein is no longer active in the presence of an input protein.

It has been suggested that biology is moving towards a more modular perspective of analyzing how those proteins and genes interact to produce a higher function [26]. Cellular behaviour is carried out and regulated by 'modules' made up of many species of interacting biomolecules to perform specific functions. These modules can be classified by function, such as genetic switches, flip-flops, logic gates, amplifiers and oscillators; or by 'network motifs' to represent interconnections that are more commonly found, such as feed-forward loops, single-input modules (SIM), and dense overlapping regulons (DOR) [27]. General principles and mechanisms governing the behaviour of modules can be elucidated through studies of synthetic networks.

Looking only at modular components of biological networks is still insufficient for understanding the system itself. To head towards a systems-level analysis, computational modeling approaches become even more important. Reductionism has been a dominant approach to studying biology, reducing a system into the components and attempting to re-connect those components through assumptions and approximations. However, a larger issue that cannot be addressed by reductionism is the lack in understanding of the dynamic and nonlinear behaviour of the systems, which can only be obtained by taking a holistic approach. Based on *in silico* prediction and optimization from computational models, more complex circuits can be rationally assembled from subnetworks. These larger circuits can then be used for further experimental study and implementation.

Many attempts thus far have focused on mapping and causally modeling the different components of biological networks. Hypotheses are then proposed to describe the system behaviour. Using the Boolean network model, on and off states have been used to describe the state of genes and proteins in a circuit [28, 29]. While a qualitative model may suggest general system behaviour, important quantitative details that dominate system behaviours may be left out, as limited by our understanding of and power to predict complex systems.

2.3.3 Applications of synthetic biology

Besides assembling synthetic networks to help advance systems biology, these networks can be used to monitor and control cellular behaviours. This includes using synthetic networks as biosensors in a natural biological system. It may be possible to supplement or replace an existing biological function in diseased cells, including the re-engineering of viral regulatory networks in the development of oncolytic viral vectors to target cancer cells [30, 31]. The kinetics of an assembled network can be tweaked for particular purposes based on computational modeling predictions, which is easier with synthetic networks than natural ones. Detailed models of synthetic systems will provide insights into drug discovery, such as revealing the effects of feedback mechanisms that may offset the effective dose of drugs [32].

Synthetic biology is also related to molecular computation research, where biological molecules can act as analogues of silicon-based integrated circuits. Computation with biological molecules has begun to surface in literature, such as molecular-based logic circuitry [33]. Modular synthetic networks can function as logic gates, and the combination

of these logic gates into higher complexity systems will feed into the study of computational devices relying on proteins and their interactions [25, 34]. From the construction and characterization of simpler elements such as switches and logic gates, it is possible to build more elaborate devices to perform higher-level functions such as memory devices as will be explored in this thesis.

2.4 Quantitative computational modeling approaches

From the above examples and applications, it is evident that quantitative simulations will be tremendously useful in both systems and synthetic biology. Conventional methods of creating network models involve performing a series of experiments, identifying specific interactions, conducting extensive literature research for confirmation, and repeating. Several methods are available to reveal regulatory relationships based solely on mRNA expression data from microarray studies. However, the many mechanisms occurring in a single system including post-transcriptional and post-translational modifications cannot be incorporated all at once on a microarray without losing precision and accuracy. A more reliable means of incorporating many different mechanisms that occur simultaneously in a system is by *in silico* simulations and modeling.

Simulations predict the kinetics of systems, incorporating assumptions and approximations to complete the models. They are generally based on statistical considerations, the validity of which can be tested experimentally. Simulations often require the integration of multiple hierarchies of models that span several orders of magnitude in terms of scale, abundance, binding affinities, and rate constants [35]. Advances in software and computational power

has allowed for more realistic, complex biological models including those for bifurcation of the cell cycle, metabolic analysis, and oscillatory circuits [36-39]. (Table 2-1) However, the choice of what can be modeled is still limited by the availability of biological knowledge.

Table 2-1: Software tools developed for the modeling and simulation of biological interactions

Tools	***Description***	***Source***
BioJake	Visualization tool for the manipulation of metabolic pathways	[40]
BioSPICE	Software system for access to current computational tools	[41, 42]
CellDesigner	Software for diagrammatic editing of biological networks	[43]
CellWare	Integrative multi-algorithmic simulation tool for deterministic and stochastic cellular events	[44, 45]
COPASI	Platform-independent tool for the simulation of biochemical events	[46]
Dizzy	Software tool for modeling integrated large-scale networks deterministically and stochastically	[47]
Dynetica	Simulation tool for studying kinetic models of dynamic networks	[48]
E-CELL	Software environment for simulation of integrative models of cellular behaviour	[49]
Gepasi	Software system for modeling chemical and biochemical reaction networks	[50-52]
Pathway Tools	Software environment for creating model-organism databases	[53]
StochSim	Stochastic simulation tool for chemical reactions	[54]
STOCKS	Stochastic kinetic simulation tool for biochemical processes	[55]

Systems Biology Workbench (SBW)	Software framework for communication between software applications	[56]
Virtual Cell	Computational framework for modeling and testing biological networks	[57, 58]

Previous studies based on quantitative models have mainly been on small and simple networks. The reasons for this restriction are a lack of complete quantitative data to input as parameters, only partial characterization of the networks by experimental studies, and expensive computational complexity required to simulate network behaviours. However, the future for quantitative approaches appears promising as better experimental procedures, including high-throughput, large-scale techniques such as microarray and mass spectrometry are developed [59, 60]. Databases are also being developed to collect shared, published experimental parameter data [61]. (Table 2-2)

Table 2-2: Databases developed to store, categorize, and share data from biological studies and modeling

Database	***Description***	***Source***
Alliance for Cellular Signaling (AfCS)	Collection of databases and tools to study signaling processes	[62]
BioModels	Database of published, peer-reviewed, quantitative models of biochemical and cellular networks	[63]
BioSilico	Integrated web-based database system for metabolic pathways	[64]
BRENDA	Information system on enzyme properties and functions	[65]
EcoCyc	Pathway database describing biological networks of *E. coli*	[66]

ENZYME	Repository for enzyme nomenclature	[67]
Kyoto Encyclopedia of Genes and Genomes (KEGG)	Knowledge database system for analysis of gene functions and pathways; includes the databases: GENES, PATHWAY, and LIGAND	[68]
MetaCyc	Database describing metabolic pathways in model organisms	[69]
ERGO (WIT)	Database system for comparative analysis of sequenced genomes and metabolic reconstructions	[70]

2.4.1 Deterministic chemical kinetics approach

Chemical kinetics represent how concentrations of each molecular species evolve in time. The interactions between molecular species are written in chemical reactions with specific rate constants and equilibrium constants. A set of chemical reactions can be represented by differential equations involving their reaction rates. The concentration changes over time of the molecular species are solved in order to produce a system transition path. A deterministic approach assumes a predictable process governed by the set of differential equations and the initial conditions [71]. Networks representing bacterial chemotaxis and bacteriophage infections have been modeled using this chemical kinetics approach and have been verified experimentally [16, 27, 72-74].

Depending on the amount of prior knowledge, many approximations are incorporated into deterministic models such as the vast amount of kinetic rates and binding affinities. Models incorporating assumed kinetic rates are useful for approximating system behaviour and function even though exact kinetic rates are not readily available. In other cases, relative

kinetic rates have been used to approximate system behaviour. Since exact kinetic rates of the components were unknown, randomly chosen kinetic rates were assigned to different components and a large number of computational models were generated [75]. Experimentation was then used to fit the predicted data. Employing this brute force approach, one group followed the time courses of major cyclin-dependent kinase activities in budding yeast cell cycles [76]. Quantitative simulations were then required to predict relevant parameters.

2.4.2 Stochastic kinetics approach

While some biological processes can be modeled using simple chemical kinetics and deterministic approaches, many are more realistically represented by random events, in which case stochastic considerations are used instead of specifying differential equations. To capture probabilistic fluctuations in gene expression and genetic regulatory networks such as that in the lysis-lysogeny decision circuit of bacteriophage lambda, stochastic approaches provided more accurate representations [77-79]. Stochastic models can also be used to deduce the effects of noise within a synthetic network, potentially leading to the manipulation of the network itself in order to improve the signal-to-noise ratio within these networks [80, 81].

A stochastic approach regards changes over time as random and unpredictable processes, with no set of differential equations defined, and takes into account inherent fluctuations that are not considered in the deterministic kinetic approach. Stochastic effects are significant in some biological systems with small molecular populations involved. Although stochastic

processes build more accurate models, they are difficult to solve analytically. However, numerical simulations are possible using Monte Carlo principles [71, 82, 83]. Instead of considering reaction parameters as reaction rates, they are treated as reaction probabilities. A software called STOCKS (STOChastic Kinetic Simulations) was developed to run Monte Carlo simulations of biochemical processes such as the binding of transcriptional regulators using a stochastic simulation algorithm [55].

Simulations using stochastic considerations have been reported for biological systems involving genetic and enzymatic reactions between molecular populations that were relatively small, including synthetic oscillatory networks, transcriptional regulation, and circadian rhythms [23, 84-86]. For large populations of molecular species, the predictions obtained from stochastic approaches match with deterministic ones. However, at smaller population sizes, stochastic effects become more dominant, in which case, deterministic approaches become insufficient [77, 87, 88]. Unfortunately, for many biological networks, stochastic simulations are still computationally expensive due to the huge differences in timescales of biological interactions and population sizes. Various improvements, approximations, and hybrid approaches have been presented [89-95]. In one such study, stochastic simulations were done on multi-scaled systems to study reactions occurring in three different regimes (slow, medium, and fast) as well as coupled reactions. The presented approach showed substantial improvement over using the basic stochastic simulation approach when applied to the study of expression and activity of Lac proteins in *E. coli* [96]. In another, a simple genetic circuit was modeled and simulated using a modified Gillespie algorithm with a quasi-steady-state assumption. This assumption was shown to greatly

simplify the stochastic model and to significantly reduce the computational complexity required, speeding up the algorithm [89].

Despite providing a more complete representation of biological networks, current stochastic approaches still face the challenge of dealing with several orders of magnitude in terms of scale and properties including binding affinities, specificities, and kinetic rates. In addition, very few of the existing stochastic methods deal with system behaviours with a spatial resolution. The majority of the simulations simplify their models by assuming spatial homogeneity for the molecular species. In addition, the models for molecular motion as well as reactions are not built upon physical principles, which may introduce artifacts to the simulation results. Hence, a new stochastic theory that can address the scalability and physically realistic is needed to accurately represent biochemical reaction networks.

2.5 Summary

Systems biology strives to understand the complex networks of biomolecular interactions in the cell. However, even with the existing amount of genomics and proteomics data, most of the parameters are still unknown about the networks in the cell. This makes it only possible to simulate and study small systems, and difficult to gain a system-level perspective of biological complexity. Hence, synthetic biology is introduced as an important tool to provide insight into natural systems by building simpler networks with well-characterized interactions in order to examine the effect of different parameters. However, the inherent complexity of biological networks requires the aid of computational modeling approaches to complement experimentation to provide a more complete perspective. Thus, to benefit from

the full potential of synthetic networks, significant advancement in quantitative computational modeling are required to attain a better representation of biological systems. Among quantitative modelling, deterministic approaches use differential equations to model systems that may be accurate for some, but inadequate for others as they lose important kinetics details such as molecular fluctuation and spatial distribution. Hence, stochastic approaches that provide a more complete representation of the system are desirable. However, current stochastic methods lack the description of spatial distribution of molecules as well as a sound physical model for the motion and chemical kinetics of the molecules. This research here seeks to bridge this gap.

Chapter 3

Theory development

3.1 Introduction

In this chapter, a method to accurately represent chemical reaction using Monte Carlo method is presented. Two issues must be addressed correctly: the motion of each molecule and the reaction kinetics of each molecule. Microscopically, the motion of each molecule is Brownian, a type of random walking motion. Macroscopically, the motion of an ensemble of molecules is described by diffusion theory. These two theories describe the same phenomenon at different levels. However, our Monte Carlo simulation is meant to work with microscopic individual molecules, while most experimentally measured diffusion parameters describe the macroscopic ensemble. A theory must be developed to bridge the two realms such that Monte Carlo simulation is able to reproduce the effects seen on macroscopic scales.

Chemical reactions can be divided into two types: reactions with only one reagent molecule and reactions involving more than one reagent molecules. Reactions involving one reagent molecules are defined here as dissociation reactions, such as:

$$A \xrightarrow{k_d} B_1 + B_2 + B_3 + \ldots$$

Reactions involving multiple reagent molecules are defined here as association reactions. These chemical reactions are caused by molecular collisions from a microscopic perspective. However, no three physically objects can collide at exactly the same time instance. It is, therefore, sufficient to describe all molecular collisions as collisions between only two molecules. This implies that any association reactions can be broken down as a combination of one or more association reactions with only two reagents. For instance, a reaction:

$$A + B + C \longrightarrow D$$

can be rewritten as a combination of:

$$A + B \longrightarrow I_{AB} \qquad I_{AB} + C \longrightarrow D$$
$$B + C \longrightarrow I_{BC} \qquad I_{BC} + A \longrightarrow D$$
$$C + A \longrightarrow I_{CA} \qquad I_{CA} + B \longrightarrow D$$

where I_{AB}, I_{BC} and I_{CA} are reaction intermediates, which may be an even more accurate description of the physical reality of the original reaction. Hence, this chapter will only deal with association reactions with two reagent molecules and one product molecule:

$$A + B \xrightarrow{k_a} C$$

Both dissociation and association reactions are described macroscopically by rate constants k_d and k_a, respectively. However, microscopically, each molecule must decide which other molecules to react with and when to react. Hence, reaction probabilities for each molecule are necessary for Monte Carlo simulation. The theory in this chapter will also bridge the gap between the microscopic reaction probability and macroscopic rate constant.

3.2 Random walk model of diffusion

A common measurable quantity describing the ensemble motion of molecules is the diffusion coefficient D. However, this diffusion coefficient does not describe the individual random walking motion of molecules. In order to simulate the random walk of each molecule, a method is devised here, relating it to the behaviour of macroscopic diffusion. With this theory, the spatial distribution of many individually random walking molecules should achieve identical distribution profile as described by macroscopic diffusion equations.

Microscopically, the function W describes a microscopic random walk (micro-RW) of a molecule by giving the probability density of a molecule walking to a location $\vec{r}$ (measured from the diffusion origin) in N steps of step length s_0:

$$W(\vec{r},N,s_0)=\left(\frac{3}{2\pi Ns_0^2}\right)^{3/2}\exp\left(-\frac{3\vec{r}^{\,2}}{2Ns_0^2}\right) \tag{3.1}$$

Macroscopically, diffusion is described by the diffusion equation:

$$\frac{\partial\phi(\vec{r},t)}{\partial t}=D\nabla^2\phi(\vec{r},t) \tag{3.2}$$

where φ is the density distribution of the diffusion species and D is the diffusion coefficient.

Because the micro-RW probability W and the macro-diffusion distribution φ have the same shape but only scaled differently, their relationship can be discovered by replacing φ with W under the similar operations as in (3.2):

$$\begin{aligned}\nabla^2W(\vec{r},N,s_0)&=\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2}\right)W(\vec{r},N,s_0)\\&=\left(\frac{9\left(x^2+y^2+z^2\right)}{N^2{s_0}^4}-\frac{9}{N{s_0}^2}\right)W(\vec{r},N,s_0)\end{aligned} \tag{3.3}$$

In micro-RW, the number of steps N is analogous to time t in macro-diffusion, hence the following partial differentiation was performed:

$$\frac{\partial}{\partial N}W(\vec{r},N,s_0)=\frac{1}{6}{s_0}^2\left(\frac{9\left(x^2+y^2+z^2\right)}{N^2{s_0}^4}-\frac{9}{N{s_0}^2}\right)W(\vec{r},N,s_0)$$

$$\therefore \frac{\partial}{dN} W(\vec{r}, N, s_0) = \frac{{s_0}^2}{6} \nabla^2 W(\vec{r}, N, s_0) \tag{3.4}$$

We introduce the average time spent for each RW step τ_{rw} by:

$$t = N\tau_{rw} \tag{3.5}$$

$$\text{hence, } \tau_{rw} = \frac{\partial t}{\partial N} \tag{3.6}$$

Compare (3.4) with (3.2) using (3.6), we get:

$$\begin{aligned} \frac{\partial}{\partial t} W(\vec{r}, N, s_0) &= \frac{\partial N}{\partial t} \frac{\partial}{\partial N} W(\vec{r}, N, s_0) \\ &= \frac{s_0^2}{6\tau_{rw}} \nabla^2 W(\vec{r}, N, s_0) \end{aligned} \tag{3.7}$$

Microscopic and macroscopic diffusion behaviour can be related by comparing (3.7) to (3.2), giving:

$$D = \frac{{s_0}^2}{6\tau_{rw}} \tag{3.8}$$

This turns out to be the 3D version of Einstein diffusion equation. The Einstein diffusion equation for 1D takes the form:

$$D = \frac{{s_0}^2}{2\tau_{rw}} \tag{3.9}$$

To interpret the physical meanings of the 3D Einstein diffusion equation, suppose *v* is the average velocity during the collision free path, (3.8) can be rewritten as:

$$v = \frac{s_0}{\tau_{rw}} \tag{3.10}$$

$$D = \frac{{s_0}^2}{6\tau_{rw}} = \frac{v^2\tau_{rw}}{6} \tag{3.11}$$

τ_{rw} is cancelled out by combining (3.5), (3.10) and (3.11):

$$N{s_0}^2 = 6Dt$$

With this relation, micro-RW $W(r, N, s_0)$ can be rewritten as $W(r, t, D)$:

$$W(\vec{r}, t, D) = \left(\frac{1}{4\pi Dt}\right)^{3/2} \exp\left(-\frac{\vec{r}^2}{4Dt}\right) \tag{3.12}$$

$W(r, t, D)$ is the diffusion probability density function (dPDF) of the end positions after a random walk using macroscopic parameters time, t and diffusion coefficient D (Figure 3-1). The time evolution of the $W(r, t, D)$ is displayed in Figure 3-2.

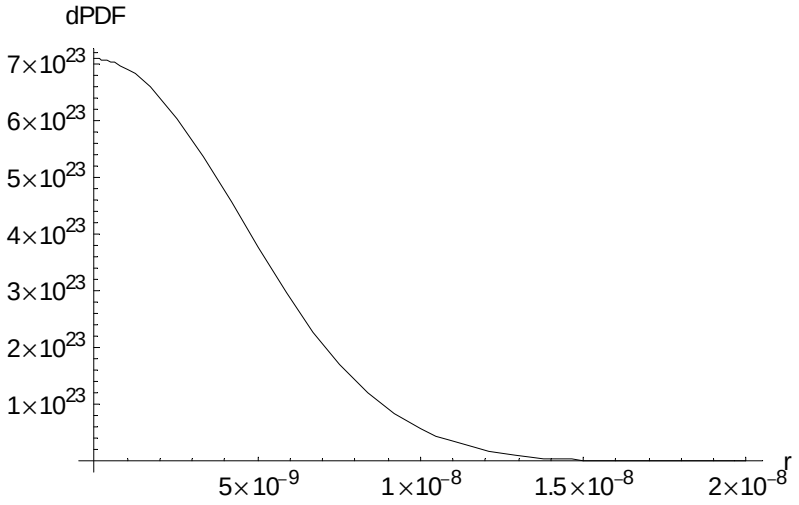

Figure 3-1: The diffusion probability density function. The x-axis is the radial distance from the origin of diffusion. Plotted with parameter: D=10^{-10}m^2/s, t=100ns. The graph assumes the shapes of a Gaussian function. From this graph, the diffusion can be estimated to be in the order of 10nm, roughly 1% of the size of a cell.

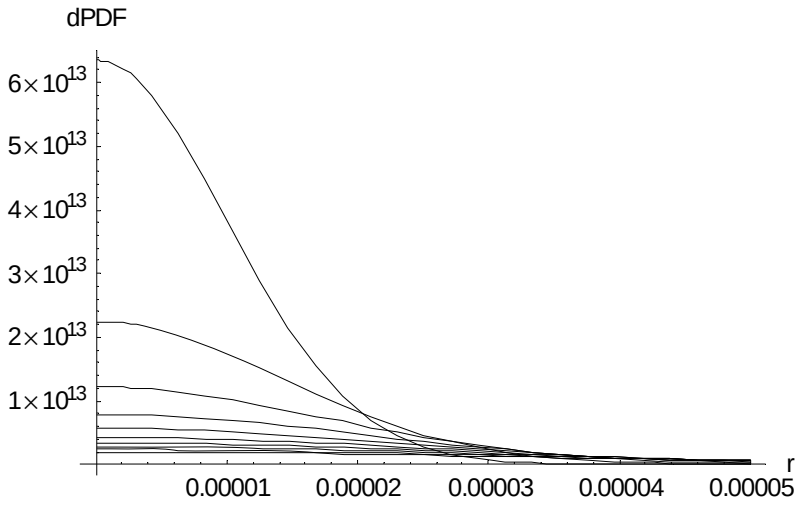

Figure 3-2: The time-evolution of the diffusion probability density function. The x-axis is the radial distance from the origin of diffusion. Plotted with parameter D=10^{-10}m^2/s, the time ranges from t=0.5s to t=5 s at a step of 0.5 s.

The probability of finding the particle in a particular volume V in 3D space is the spatial integration of the probability density function over that volume:

$$P = \iiint_{\vec{r} \in V} W(\vec{r}, t, D) dr \tag{3.13}$$

W(r, t, D) is normalized, verifying that the probability of finding a particle in the entire 3D space is 1:

$$\iiint_{\vec{r} \in \mathbb{R}^3} W(\vec{r}, t, D) dr = 1 \tag{3.14}$$

W(r, t, D) is also superimposible, which means that if the diffusion density distribution from a point source after $\Delta t_1$is *W(r,* Δt_1*, D)*, and then every point is treated as new point source and diffused for Δt_2, the resulting diffusion density distribution is identical to the one diffusing for $\Delta t_1 + \Delta t_2$. Hence, convoluting *W(r,* Δt_1*, D)* with *W(r,* Δt_2*, D)* results in *W(r,* $\Delta t_1 + \Delta t_2$*, D)*:

$$W(\vec{r},\Delta t_1,D)\otimes W(\vec{r},\Delta t_2,D)=W(\vec{r},\Delta t_1+\Delta t_2,D) \tag{3.15}$$

Furthermore:

$$W(\vec{r},\Delta t_1,D)\otimes W(\vec{r},\Delta t_2,D)\otimes ...\otimes W(\vec{r},\Delta t_n,D)=W(\vec{r},\sum_i \Delta t_i,D) \tag{3.16}$$

This implies that using this *W(r, t, D)* function, diffusion can be divided into arbitrary time intervals, in each of which diffusion can be independently simulated to yield the same diffusion as if diffusion were carried out in one step.

3.3 Simulation of random walk

In diffusion simulation, the density distribution after the same total duration should be the same regardless to the choice of time step duration Δt of the simulation. Hence, it poses a challenge to assign the random walk vector of each molecule give Δt such that the behaviour of individual molecules correctly contributes to statistical distribution of all molecules. An accurate Monte Carlo method describing molecular diffusion is described in the following text.

3.3.1 Random number generation with non-uniform distribution

First, a method to generate random numbers with non-uniform distribution is described. The application of this method to our simulation will become evident in the following sections. Random numbers are generally used in Monte Carlo simulations to provide sample values of random variables such as thermal noise, random forces, etc.. Most random number generators give uniform distribution of numbers, i.e. all numbers in a defined range have equal probability of being chosen. This may not always be the case for random parameters in

simulations. For instance, to describe the normal distribution of test marks of a class, a random number generator biasing towards certain range of mark must be used. Here a method is devised to provide random number with arbitrary distribution profile: $P(x)$ of x (the random variable), $x \in [x_1, x_2]$.

The distribution of x is analogous to the histogram of x. Hence, if the area under $P(x)$ between $x \in [x_1, x_2]$ is sampled uniformly, the value of x for each sample point would have a sampling probability identical to $P(x)$ (Figure 3-3). The procedure can be carried out as follows:

1. Generate a random number $R \in \left[0, \int_{x1}^{x2} P(x)dx\right]$, where R comes from uniform random number generator
2. Solve for x in $R = \int_{x1}^{x} P(x)dx$:

   Suppose the integration is:

   $$F(x) = \int P(x)dx$$

   $R = \int_{x1}^{x} P(x)dx$ then becomes:

   $$R = \int_{x1}^{x} P(x)dx = F(x) - F(x_1)$$

   $$F(x) = R + F(x_1)$$

   hence,

   $$x = F^{-1}\left(R + F(x_1)\right) \tag{3.17}$$

3. Using this equation, x would then have the $P(x)$ distribution in $x \in [x_1, x_2]$

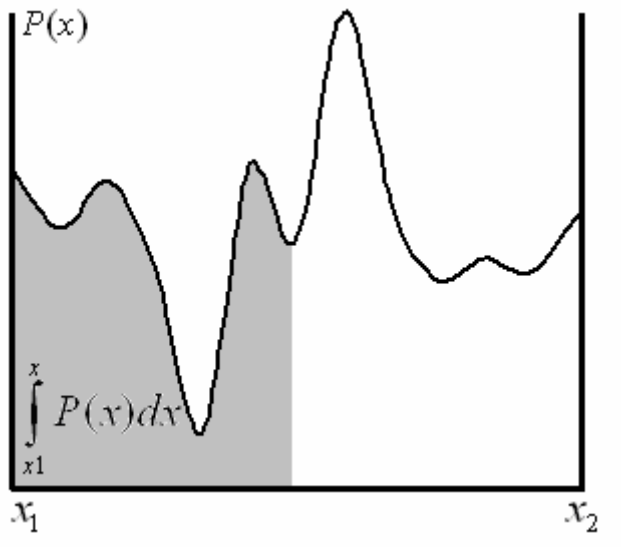

Figure 3-3: Probability distribution function of arbitrary shape integrated from x_1 to x. The x-axis is the desired random number, the y-axis is the probability distribution of the random number x.

3.3.2 Random 3D-isotropic sampling:

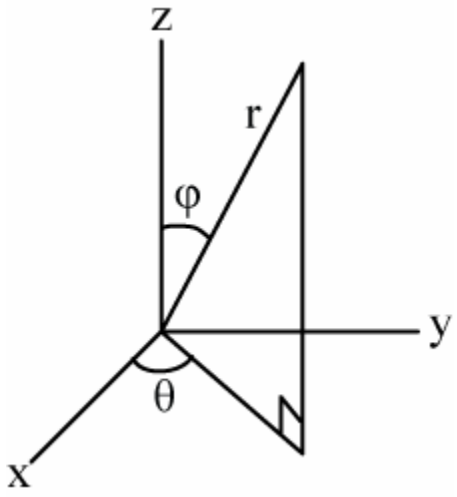

Figure 3-4: Spherical coordinate system, *r*, *θ*, and *φ* are defined as shown.

To achieve isotropic sampling in all directions, θ is uniformly sampled from 0 to 2π. Hence, the probability distribution of θ is a constant function: $P(\theta) \propto 1$ with $\theta \in [0, 2\pi)$. However,

if ϕ were uniformly sampled from 0 to π, it leads to a sampling density distribution concentrated at the polar region with a density distribution of $1/\sin\phi$ (see Figure 3-5). To correct for this anisotropic distribution, more samples of ϕ must be taken near the equator than at the poles. The probability distribution of ϕ is therefore not a constant function of ϕ but rather: $P(\phi) \propto \sin\phi$ with $\phi \in [0, \pi]$.

random sampling probability of θ and ϕ are: $\begin{matrix} P(\theta) \propto 1, \ \theta \in [0, 2\pi) \\ P(\phi) \propto \sin\phi, \ \phi \in [0, \pi] \end{matrix}$

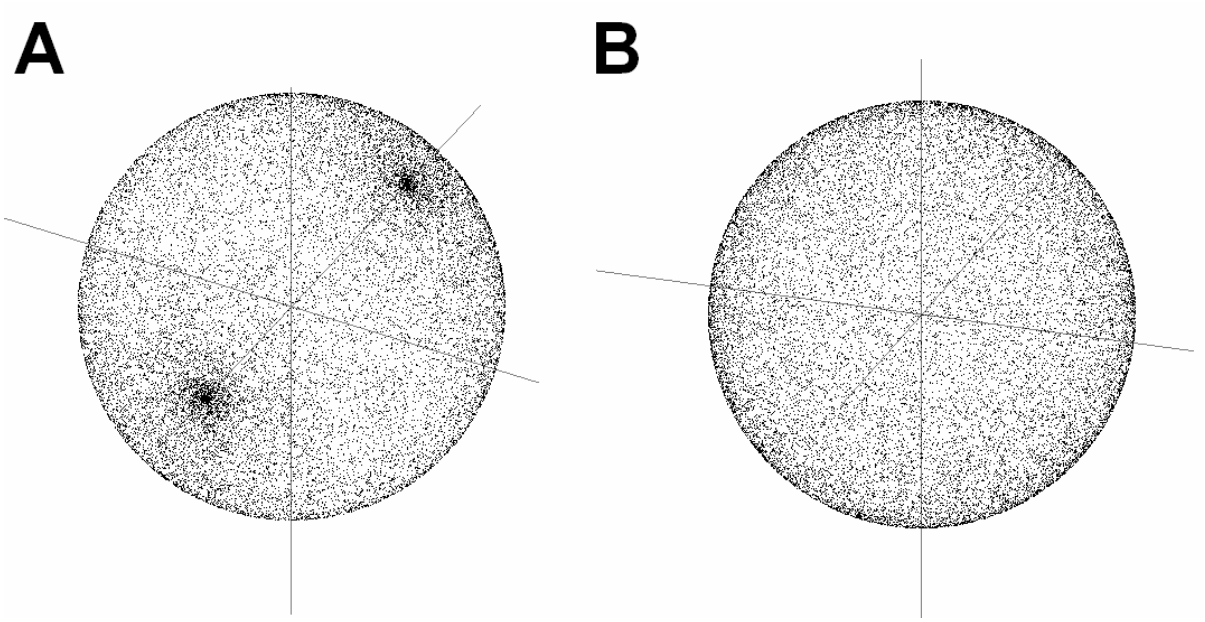

Figure 3-5: Comparison of: A. uniform random sampling of θ and ϕ and B. corrected random sampling of θ and ϕ. The uniform sample of θ and ϕ causes higher density at the polar regions and low density at the equator, hence, the distribution is not isotropic.

Applying (3.17) to $P(\phi)$, $\phi \in [0, \pi]$:

$$
\begin{aligned}
R &= \int_0^{\phi} \sin\phi' \, d\phi' \\
&= \left(-\cos\phi'\right)\Big|_0^{\phi} \\
&= 1 - \cos\phi \\
\phi &= \arccos\left(1 - R\right), \ \phi \in [0, \pi], \ R \in [0, 2]
\end{aligned}
$$

where R is a random number between 0 and 2. Hence, to generate isotropy using spherical coordinate:

$$
\begin{aligned}
\theta &= rand, \quad rand \in [0, 2\pi] \\
\phi &= \arccos(1 - rand), \quad rand \in [0, 2]
\end{aligned}
\tag{3.18}
$$

The directional distribution using this set of equation is isotropic (Figure 3-5).

3.3.3 Diffusion step size

Equation (3.12) shows the diffusion probability density function. Being a density function, it does not reflect the actual number of molecules within each shell surrounding the diffusion centre. Consider what is depicted in Figure 3-6, the density is highest in the centre, however the population is highest at some distance from the centre. To generate the correct density distribution given by *W(r, t, D)*, the sampling probability for each radius *r* must be proportion to the population distribution (1D radial density distribution) and not the 3D density distribution.

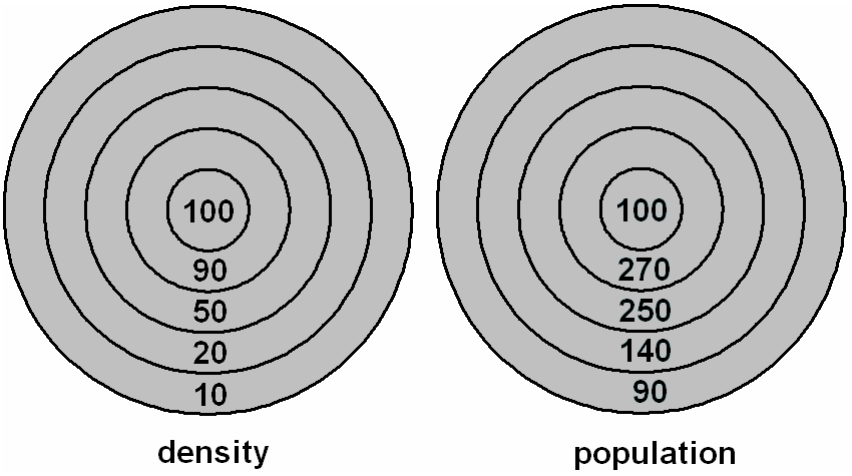

Figure 3-6: Comparison of the density distribution and population distribution in 2D. The numbers shown in each ring indicate the unit density / population within that ring. The population within each ring is then the area of that ring multiplied by the density in that ring. If the density distribution peaks at the centre as in a Gaussian distribution, the population would peak at some distance away from the centre.

The population (1D radial density) distribution is given by integrating *W(r, t, D)* over spherical shells:

$$\begin{aligned} P(r,t,D) &= \int_0^{\pi} r d\phi \int_0^{2\pi} r \sin\phi W(\vec{r},t,D) d\theta \\ &= W(r,t,D) r^2 \int_0^{\pi} d\phi \sin\phi \int_0^{2\pi} d\theta \quad \text{since } W \text{ is spherically symmetrical} \\ &= 4\pi r^2 W(r,t,D) \end{aligned}$$

We verify again that *P(r)* is normalizable:

$$\begin{aligned} \int_0^{\infty} P(r,t,D) dr &= \int_0^{\infty} dr \int_0^{\pi} r d\phi \int_0^{2\pi} r \sin\phi W(r,t,D) d\theta \\ &= 1 \end{aligned}$$

$$
\begin{aligned}
R(r) &= \int_0^r P(r')dr' \\
&= \int_0^r 4\pi r'^2 W(r',t,D)dr' \\
&= 4\pi\left(\frac{1}{\sqrt{4\pi Dt}}\right)^3 \int_0^r r'^2 \exp\left(-\frac{r'^2}{4Dt}\right)dr' \\
&= 4\pi\left(\frac{1}{\sqrt{\pi}}\right)^3 \int_0^r \frac{r'^2}{4Dt}\exp\left(-\frac{r'^2}{4Dt}\right)d\frac{r'}{\sqrt{4Dt}} \\
&= \frac{4}{\sqrt{\pi}}\int_0^{\frac{r}{\sqrt{4Dt}}} x^2\exp\left(-x^2\right)dx \\
&= \frac{4}{\sqrt{\pi}}\left(-\frac{1}{2}\right)\int_0^{\frac{r}{\sqrt{4Dt}}} dx\exp\left(-x^2\right) \\
&= \frac{4}{\sqrt{\pi}}\left(-\frac{1}{2}\left(x\exp\left(-x^2\right)\right)\Big|_0^{\frac{r}{\sqrt{4Dt}}} + \frac{1}{2}\underbrace{\int_0^{\frac{r}{\sqrt{4Dt}}}\exp\left(-x^2\right)dx}_{\frac{1}{2}\sqrt{\pi}erf\left(\frac{r}{\sqrt{4Dt}}\right)}\right) \\
&= \frac{4}{\sqrt{\pi}}\left(-\frac{1}{2}\left(\frac{r}{\sqrt{4Dt}}\exp\left(-\frac{r^2}{4Dt}\right)\right)+\frac{1}{4}\sqrt{\pi}erf\left(\frac{r}{\sqrt{4Dt}}\right)\right) \\
&= erf\left(\frac{r}{\sqrt{4Dt}}\right)-\frac{r}{\sqrt{\pi Dt}}\exp\left(-\frac{r^2}{4Dt}\right)
\end{aligned}
\tag{3.19}
$$

The inverse function *r(R)*, diffusion radius *r* as a function of random number *R*, was found numerically by solving (3.19) at each *R* from 0 to 1 with 0.0001 intervals. (Figure 3-7)

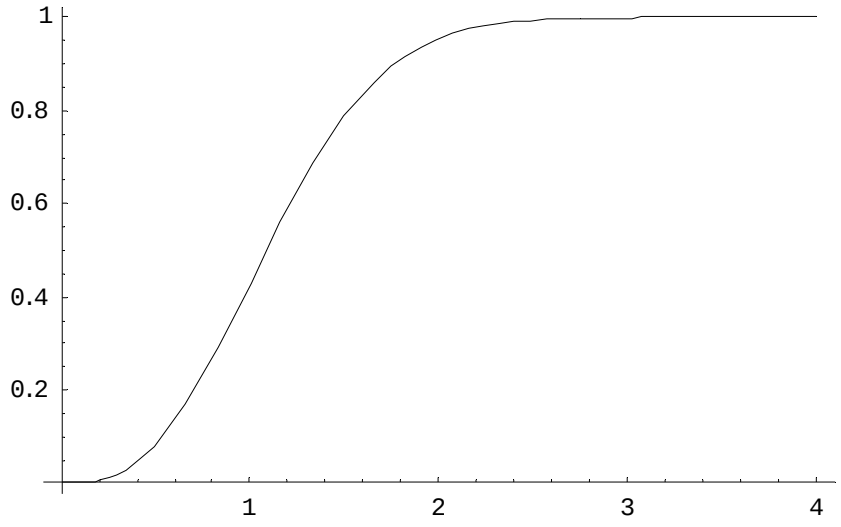

Figure 3-7: Plot of *R(r)*. Horizontal axis is the radial distance $\frac{r}{\sqrt{4Dt}}$ in spherical coordinate, the vertical axis is *R*, indicating that if the variable *R* were to be random sampled, the radial distribution will achieve Gaussian density distribution.

3.4 Molecular collisions from random walk

In this section, the interaction, in particular, the collision between two random walking molecules is studied. One parameter to consider is the size of the molecules. First, imagine two elephants random walking from 10 meters apart, their likelihood to collide is high. Imagine again the same setup but with two flies. The probability of the flies flying into each other is much lower. Going into the extreme of two ideal points, given the same period of time, it is nearly impossible for them to collide. Intuitively, it suggests that the larger the size of random walking objects, the higher their collision probability is in a given time duration. Similarly, the sizes of molecules are important for their collision rate.

Two molecules A and B with finite size, their collision can be defined as their centre of mass being within certain distance. Assuming they are rigid spheres, the distance is the sum of the radius of the two spheres. This may be a rough estimation, but it allows a simple model to be built upon it. Later in this section, a factor will be introduced to correct for this assumption. Therefore, if A and B both walk into a spherical volume with diameter equal to the sum of their radii, a collision occurs.

To simplify the math, molecule A and B are placed on the x-axis with a separation of l_0, the origin is placed at the midpoint of A and B. The diffusion probability density function (dPDF) for molecule A and B are W_A and W_B, as defined in the previous section, are:

$$W_A\left(\vec{r},t,D_A\right)=\left(\frac{1}{4\pi D_A t}\right)^{3/2}\exp\left(-\frac{\left(x+\frac{1}{2}l_0\right)^2+y^2+z^2}{4D_A t}\right)$$

$$W_B\left(\vec{r},t,D_B\right)=\left(\frac{1}{4\pi D_B t}\right)^{3/2}\exp\left(-\frac{\left(x-\frac{1}{2}l_0\right)^2+y^2+z^2}{4D_B t}\right)$$

An effective collision volume V_c is defined here to describe the volume within which collision occurs:

$$V_c = V_0 + V_f \tag{3.20}$$

It is the sum of two volumes: where V_0 is the collision volume strictly defined by the geometry of the molecules and V_f is a correction "fictitious" volume. Within this "fictitious" volume, the molecule pair feel either repletion (-) or attraction (+) force between them, which may increase or decrease the likelihood of collision, hence changing the effective collision volume. For instance, if there is attraction between a pair of molecules, the fate of collision between the pair may be determined when they haven't physically collided. In the contrary,

if two molecules repel each other, even though they get close by, their chance of collision is decreased by the force pushing them apart. Therefore, the size of the molecules alone is not a good indicator of the collision volume.

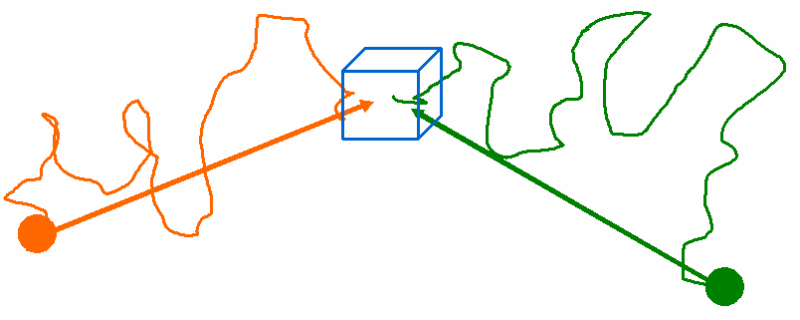

Figure 3-8: Illustration of random walk collision. The wiggly lines are the actual path each molecule takes to get to the collision site. The arrows indicate the vector from molecule original location to the collision site, which gives the probability they will land there. The box is the volume in which collision would occur.

Because the collision volume is relatively small compared to the molecular distance, W_A and W_B values can be assumed as constants in the collision volume V_c. Hence, the probability of A and B walking into the volume V_c are:

$$\iiint_{\vec{r}\in Vc} W_A(\vec{r},t,D_A)d\vec{r} \approx W_A(\vec{r},t,D_A)\iiint_{\vec{r}\in Vc} d\vec{r}$$
$$= W_A(\vec{r},t,D_A)V_c$$

$$\iiint_{\vec{r}\in Vc} W_B(\vec{r},t,D_B)d\vec{r} \approx W_B(\vec{r},t,D_B)\iiint_{\vec{r}\in Vc} d\vec{r}$$
$$= W_B(\vec{r},t,D_B)V_c$$

Therefore, the collision probability inside a volume of V_c, at ***r*** from the origin and after a period of t is:

$$P_c(\vec{r},t,l_0)=\iiint_{\vec{r}\in Vc} W_A(\vec{r},t,D_A)d\vec{r}\iiint_{\vec{r}\in Vc} W_B(\vec{r},t,D_B)d\vec{r}\approx W_A(\vec{r},t,D_A)W_B(\vec{r},t,D_B){V_c}^2 \tag{3.21}$$

It follows that the collision probability density function (cPDF) $\rho_c(\vec{r},t,l_0)$ is:

$$\begin{aligned}
\rho_c(\vec{r},t,l_0)&=\frac{P_c(\vec{r},t,l_0)}{V_c}\approx W_A(\vec{r},t,D_A)W_B(\vec{r},t,D_B)V_c\\
&=\left(\frac{1}{4\pi\sqrt{D_AD_Bt}}\right)^3\exp\left(-\frac{(D_A+D_B)\left(x^2+y^2+z^2+\frac{1}{4}{l_0}^2\right)+(D_A-D_B)l_0x}{4D_AD_Bt}\right)V_c\\
&=\left(\frac{1}{4\pi\sqrt{D_AD_Bt}}\right)^3\exp\left(-\frac{(D_A+D_B)}{4D_AD_Bt}\left(\begin{array}{l}x^2+\dfrac{(D_A-D_B)}{(D_A+D_B)}l_0x+\left(\dfrac{(D_A-D_B)}{2(D_A+D_B)}l_0\right)^2\\ \qquad +y^2+z^2+\dfrac{1}{4}\left(1-\dfrac{(D_A-D_B)^2}{(D_A+D_B)^2}\right){l_0}^2\end{array}\right)\right)V_c\\
&=\left(\frac{1}{4\pi\sqrt{D_AD_Bt}}\right)^3\exp\left(-\frac{(D_A+D_B)}{4D_AD_Bt}\left(\begin{array}{l}\left(x+\dfrac{(D_A-D_B)}{2(D_A+D_B)}l_0\right)^2+y^2+z^2\\ \qquad +\dfrac{1}{4}\left(1-\dfrac{(D_A-D_B)^2}{(D_A+D_B)^2}\right){l_0}^2\end{array}\right)\right)V_c
\end{aligned} \tag{3.22}$$

The time evolution of the collision probability density function as well as its spatial distribution is shown in Figure 3-9 and Figure 3-10.

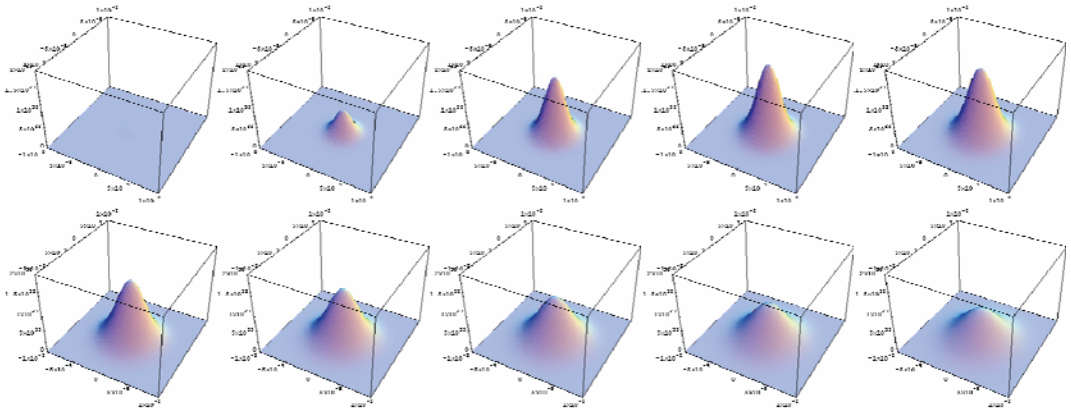

Figure 3-9: The time evolution of collision probability density in 2D. The horizontal plane indicates the physical space where molecules move and collide. The vertical axis is the collision probability density.

The parameters used here are: D_A=D_B =10^{-10} m^2/s, Protein diameter=5 nm, V_c=5.2e-25 m^3, l_0=10 nm, time range=[10ns, 100ns] at 10ns interval. The peak is located at the centre of the two reagent molecules.

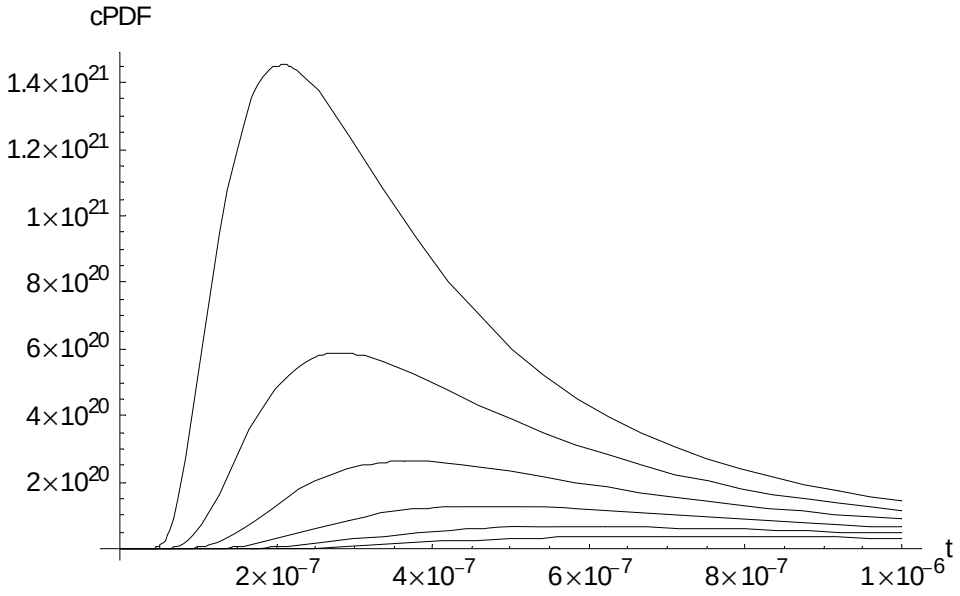

Figure 3-10: Spatial variation of the collision probability density function. The horizontal axis is time, the vertical axis is the collision probability function. Each curve shows the cPDF at a distance from the maximum location of the cPDF function (from 10 nm to 20 nm at 2 nm intervals). Parameters used in this plotting are: D_A=D_B =10^{-10} m^2/s, Protein diameter=5 nm, V_c=5.2e-25 m^3, l_0=10 nm.

Integrating the cPDF over the entire 3D space at time t gives the probability of collision $P_c(t, l_0)$:

$$
\begin{aligned}
P_c(t,l_0) &= \iiint_{\vec{r}\in\mathbb{R}} \rho_c(\vec{r},t,l_0)d\vec{r} \\
&= \left(\frac{1}{4\pi\sqrt{D_A D_B}t}\right)^3 V_c \exp\left(-\frac{(D_A+D_B)}{4D_A D_B t}\frac{1}{4}\left(1-\frac{(D_A-D_B)^2}{(D_A+D_B)^2}\right)l_0{}^2\right) \\
&\qquad \iiint_{\vec{r}\in\mathbb{R}} \exp\left(-\frac{(D_A+D_B)}{4D_A D_B t}\left(\left(x+\frac{(D_A-D_B)}{2(D_A+D_B)}l_0\right)^2+y^2+z^2\right)\right)d\vec{r} \\
&= \left(\frac{1}{4\pi\sqrt{D_A D_B}t}\right)^3 V_c \exp\left(-\frac{l_0{}^2}{4(D_A+D_B)t}\right)\iiint_{\vec{r}\in\mathbb{R}} \exp\left(-\frac{(D_A+D_B)}{4D_A D_B t}\left(x^2+y^2+z^2\right)\right)d\vec{r}
\end{aligned}
$$

(because integration is over the entire $\mathbb{R}$, we can shift the entire coodinate in the x-direction by $\frac{(D_A-D_B)}{2(D_A+D_B)}l_0$, thus making the exp(x) term disappear)

$$
\begin{aligned}
&= \left(\frac{1}{4\pi\sqrt{D_A D_B}t}\right)^3 V_c \exp\left(-\frac{l_0{}^2}{4(D_A+D_B)t}\right)\left(\int_{-\infty}^{+\infty}\exp\left(-\frac{(D_A+D_B)}{4D_A D_B t}x^2\right)dx\right)^3 \\
&= \left(\frac{1}{4\pi\sqrt{D_A D_B}t}\right)^3 V_c \exp\left(-\frac{l_0{}^2}{4(D_A+D_B)t}\right)\sqrt{\frac{\pi}{\frac{(D_A+D_B)}{4D_A D_B t}}}^3 \\
&= \left(\frac{1}{4\pi(D_A+D_B)t}\right)^{3/2} V_c \exp\left(-\frac{l_0{}^2}{4(D_A+D_B)t}\right)
\end{aligned}
$$

(3.23)

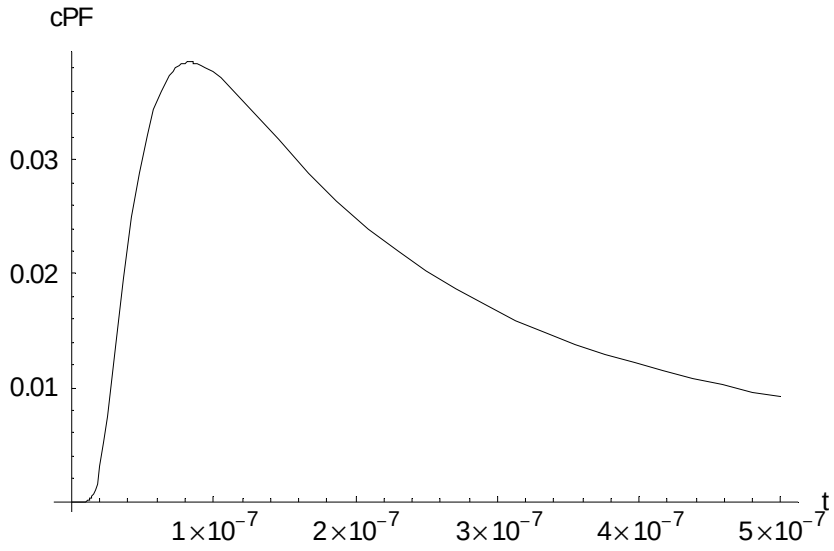

Figure 3-11: Collision probability function with respect to time. The following parameters are used: D_A=D_B =10^{-10} m^2/s, protein diameter=5 nm, V_c=5.2e-25 m^3, l_0=10 nm.

This function describes the collision probability at time *t*. The two molecules are initially separated at a distance l_0 and allowed to diffuse at *t*=0 with diffusion constant D_A and D_B. Note that, this is not the probability of collision since *t*=0, but the probability at the instance of *t*.

The general features on this curve are expected: the collision probability remains 0 until shortly after t=0 as diffusion must occur before the molecules can travel far enough to collide with each other; soon after this, the probability would reach a maximum; and then, as time goes on, the statistical distance between A and B would increase due to diffusion, hence the decreasing collision probability.

3.5 Association reactions from a microscopic perspective

Reactions are direct results of molecular collisions. During the development of this theory, it was realized that the collision probability alone is not sufficient to provide the reaction probability. Imagine two physically identical situations simulated with different time step durations: using small time step durations, if a pair of molecules collided, the same pair of molecules could stay in collision state for many time steps thereafter, as it would take many steps of diffusion to bring them out of the collision distance. However it could take as few as one step if the time step duration were significantly longer. If the reaction probability is directly proportional to the collision probability, then smaller time step duration would give rise to a much higher reaction probability. Hence, the probability of reaction per time step not only dependents on the collision probability $P_c(t, l_0)$, but also on the length of the time step duration. In other words, whether a reaction occurs depends on how long two molecules stay in the collision state. Therefore, an accumulative collision probability function (cAPF) is introduced:

$$\int_0^t P_c(t',l_0)dt' = \int_0^t \left(\frac{1}{4\pi\left(D_A+D_B\right)t'}\right)^{3/2} V_c \exp\left(-\frac{{l_0}^2}{4\left(D_A+D_B\right)t'}\right)dt'$$

$$= -\frac{V_c}{2\left(D_A+D_B\right)\pi^{3/2}l_0}\int_0^t \exp\left(-\frac{{l_0}^2}{4\pi\left(D_A+D_B\right)t'}\right)d\left(\frac{l_0}{\sqrt{4\left(D_A+D_B\right)t'}}\right)$$

$$= -\frac{V_c}{2\left(D_A+D_B\right)\pi^{3/2}l_0}\int_{+\infty}^{\frac{l_0}{\sqrt{4(D_A+D_B)t}}} \exp\left(-x^2\right)dx$$

$$= \frac{V_c}{2\left(D_A+D_B\right)\pi^{3/2}l_0}\left(\int_{\frac{l_0}{\sqrt{4(D_A+D_B)t}}}^{+\infty} \exp\left(-x^2\right)dx\right)$$

$$= \frac{V_c}{2\left(D_A+D_B\right)\pi^{3/2}l_0}\left(\int_0^{+\infty} \exp\left(-x^2\right)dx - \int_0^{\frac{l_0}{\sqrt{4(D_A+D_B)t}}} \exp\left(-x^2\right)dx\right)$$

$$= \frac{V_c}{2\left(D_A+D_B\right)\pi^{3/2}l_0}\left(\frac{\sqrt{\pi}}{2} - \frac{\sqrt{\pi}}{2}erf\left(\frac{l_0}{\sqrt{4\left(D_A+D_B\right)t}}\right)\right)$$

$$= \frac{V_c}{4\left(D_A+D_B\right)\pi l_0}\left(1 - erf\left(\frac{l_0}{\sqrt{4\left(D_A+D_B\right)t}}\right)\right) \qquad (3.24)$$

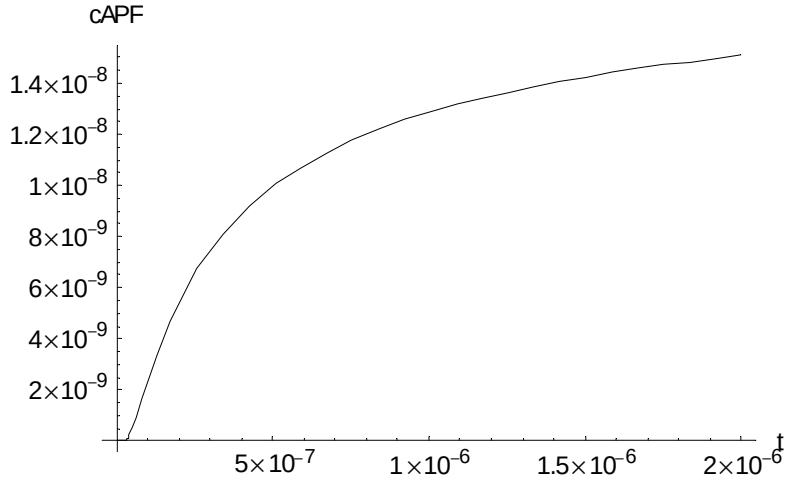

Figure 3-12: The accumulative collision probability function $\int_0^t P_c(t',l_0)dt'$ as a function of time.

This function has an asymptotical upper bound of $\frac{V_c}{4(D_A + D_B)\pi l_0}$ as time approaches infinity. The physical meaning of this function is the expected duration of time in which the two molecules remain in the collision state in a given period of time t:

$$\begin{aligned} \langle t_c \rangle &= \sum_{t'=0-t} P_c(t', l_0)\Delta t' \\ &= \int_0^t P_c(t', l_0)dt' \text{ in the limit of } \Delta t' \to 0 \end{aligned} \tag{3.25}$$

Not every collision results in reaction, to relate the probability of reaction to collisions between molecules, a hypothetical collision-reaction time constant, τ_r, is introduced here. This is the time constant by which a collision state between two molecules will "decay" into the reaction product. The reaction probability $P_r(t, l_0)$ is then:

$$P_r(t, l_0) = 1 - \exp\left(-\frac{\langle t_c \rangle}{\tau_r}\right) \tag{3.26}$$

This hypothetical formulation provides satisfies key features required of the reaction probability:

- The longer two molecules remain in collision state, the more likely they will react, hence the function must be an increasing function.
- The probability is 0 if the expected collision state duration is 0, hence the function must pass through the origin.
- The probability of reaction between two molecules cannot exceed 100%. Hence, the function must have an upper bound of 1.

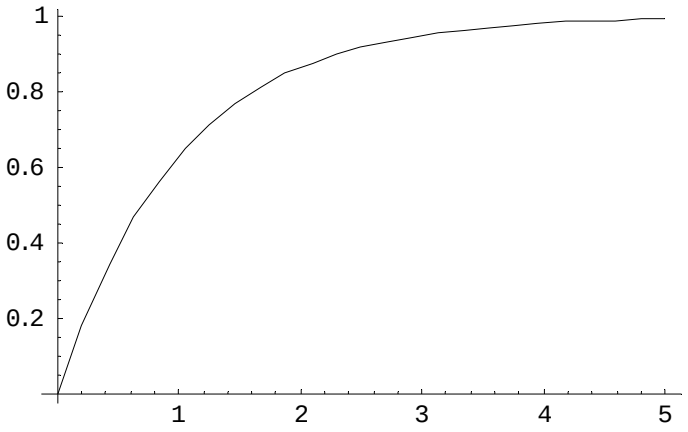

Figure 3-13: Probability of reaction as a function of the duration of collision state $<t_c>$. The horizontal axis is in unit of τ_r.

The collision-reaction time constant, τ_r, takes into account that even with the same expected collision duration, some reactions are more likely to occur than others due to difference in activation energy barrier and steric hindrance. This does not conceptually conflict with the effective collision volume (V_c), because V_c deals with forces that bring molecules into collision, whereas τ_r deals with which collision could result in reaction.

One question naturally arising from this hypothesis is whether it matters if the collision state is continuous or fragmented. For example, if the collision state occupies 50% of time, it could be a continuous 0.5 second for every 1 second (scenario 1), or 10 of 0.05 second continuous collision state for every 1 second (scenario 2). The total probability of reaction is then related to the separated collision events by:

Scenario 1: since there is only one partition in the duration of collision state, i.e. collision time is continuous:

$$P_r(t)_{total} = 1 - \exp\left(-\frac{\langle t_c \rangle_{total}}{\tau_r} \right) \tag{3.27}$$

Scenario 2: the total duration of collision state is partitioned into *n* discontinuous portions, the total probability of no reaction happening is if reaction does not happen in any of the partitions:

$$1 - P_r(t)_{total} = \left(1 - P_r(t)_1\right)\left(1 - P_r(t)_2\right)...\left(1 - P_r(t)_n\right) \tag{3.28}$$

$$\begin{aligned} P_r(t)_{total} &= 1 - \prod_{i=1}^{n} \left(1 - P_r(t)_i\right) \\ &= 1 - \prod_{i=1}^{n} \left(1 - 1 + \exp\left(-\frac{\langle t_c \rangle_i}{\tau_r} \right) \right) \\ &= 1 - \prod_{i=1}^{n} \exp\left(-\frac{\langle t_c \rangle_i}{\tau_r} \right) \\ &= 1 - \exp\left(-\frac{\sum_{i=1}^{n} \langle t_c \rangle_i}{\tau_r} \right) \end{aligned} \tag{3.29}$$

since $\sum_{i=1}^{n} \langle t_c \rangle_i = \langle t_c \rangle_{total}$ in any given time interval,

$$P_r(t)_{total} = 1 - \exp\left(-\frac{\langle t_c \rangle_{total}}{\tau_r} \right) \tag{3.30}$$

The reaction probability from scenario 2 is identical to scenario 1. Hence, we proved that only the total collision duration length matters, the details on how often they collide and how the collision state duration partitions does not matter. Substituting the expression for $<t_c>_{total}$

(3.24) and (3.25), we get reaction probability between two molecules with initial separation of l_0 and after a period of time t:

$$P_r(t,l_0) = 1 - \exp\left(-\frac{1}{\tau_r} \int_{\Delta t} P_c(t,l_0)_{A_i B_j}\, dt \right)$$
$$= 1 - \exp\left(-\frac{1}{\tau_r} \frac{V_c}{4(D_A + D_B)\pi l_0} \left(1 - erf\left(\frac{l_0}{\sqrt{4(D_A + D_B)t}} \right) \right) \right) \qquad (3.31)$$

However, the goal is to be able to use real kinetics parameters in simulations. The hypothetical parameters τ_r and V_c are not given available in literatures, and must be substituted by real kinetics parameter k_a.

3.6 Association reaction - relating microscopic and macroscopic behaviour

In this section, the question of how is the microscopic behaviour related to the association rate constant is answered. To study the problem, an ensemble of molecules must be studied instead of a single pair. Supposed there is a collection of molecule A: $\{A_i\}$; and B: $\{B_j\}$, the reaction probability of an individual pair A_i-B_j, P_r^{ij}, is:

$$P_r^{ij}(t,l_{ij}) = 1 - \exp\left(-\frac{1}{\tau_r} \int_{\Delta t} P_c(t,l_{ij})_{A_i B_j}\, dt \right)$$
$$= 1 - \exp\left(-\frac{1}{\tau_r} \frac{V_c}{4(D_A + D_B)\pi l_{ij}} \left(1 - erf\left(\frac{l_{ij}}{\sqrt{4(D_A + D_B)t}} \right) \right) \right) \qquad (3.32)$$

The reaction probability between A_i and all of $\{B_j\}$, $P_r^{\,i}$, can be related by:

$$P_r^i = 1 - \prod_j \left(1 - P_r^{ij}\right)$$

Hence,

$$\begin{aligned} P_r^i(t,\{l_{ij}\}) &= 1 - \prod_j \exp\left(-\frac{1}{\tau_r} \int_{\Delta t} P_c(t,l_{ij})_{A_i B_j}\, dt \right) \\ &= 1 - \exp\left(-\frac{1}{\tau_r} \sum_j \int_{\Delta t} P_c(t,l_{ij})_{A_i B_j}\, dt \right) \\ &= 1 - \exp\left(-\frac{1}{\tau_r} \sum_j \frac{V_c}{4\left(D_A + D_B\right)\pi l_{ij}} \left(1 - erf\left(\frac{l_{ij}}{\sqrt{4\left(D_A + D_B\right)t}} \right) \right) \right) \end{aligned} \tag{3.33}$$

The critical question of how to relate this to empirical rate constant must be answered. To link the association rate constant k_a with microscopic quantities τ_r and V_c, consider a single molecule of A in a homogeneous concentration of molecule B. Under this situation, the concentration of B is far greater than A, hence the concentration of B can be regarded as a constant B_0. The rate equation for A, it can then be written as:

$$\begin{aligned} \frac{dA(t)}{dt} &= -k_a A(t) B(t) \\ &= -k_a A(t) B_0 \\ A(t) &= A_0 \exp(-k_a B_0 t) \end{aligned} \tag{3.34}$$

The macroscopic description of reaction probability of the single molecule A is then:

$$P_r^{total}(t) = 1 - \exp\left(-k_a B_0 t\right) \tag{3.35}$$

The microscopic description of reaction probability of the single molecule A is:

$$P_r^{total}(t,\{B_j\}) = 1 - \exp\left(-\frac{1}{\tau_r} \sum_j \int_0^t P_c(t,l_{ij})_{AB_j}\, dt \right) \tag{3.36}$$

Both descriptions should yield the same result, hence equating them gives:

$$
\begin{aligned}
1-\exp\left(-k_a B_0 t\right) &= 1-\exp\left(-\frac{1}{\tau_r}\sum_j \int_0^t P_c(t,l_{ij})_{AB_j}\, dt\right) \\
\exp\left(-k_a B_0 t\right) &= \exp\left(-\frac{1}{\tau_r}\sum_j \int_0^t P_c(t,l_{ij})_{AB_j}\, dt\right) \\
k_a B_0 t &= \frac{1}{\tau_r}\sum_j \int_0^t P_c(t,l_{ij})_{AB_j}\, dt
\end{aligned}
\tag{3.37}
$$

The summation over all *j* means that in the infinite 3D space homogeneously filled with B, the expected collision duration of the A molecule with all B_j molecules. One can immediately see this is the equation relating k_a with our hypothetical parameters τ_r and V_c (contained in $P_c(t)$). The above summation is impossible to carry out, hence to sum over the entire 3D space with homogeneous B, we convert the summation to integration with:

$$
\begin{aligned}
\frac{\sum_{j\in\text{every point in }\mathbb{R}^3} \frac{1}{N_A}}{\int_0^\infty 4\pi l^2 dl} &= B_0 \\
\sum_{j\in\text{every point in }\mathbb{R}^3} 1 &= \int_0^\infty 4\pi B_0 N_A l^2 dl
\end{aligned}
\tag{3.38}
$$

where N_A is the Avogadro's number. The integral converts the discrete molecular distribution of $\{B_j\}$ to a continuous homogenous "field" of B such that the "density" of B in this field is the concentration of B in the discrete space. Hence:

$$
k_a B_0 t = \frac{1}{\tau_r}\int_0^{+\infty} 4\pi B_0 N_A l^2 \int_0^t P_c(t,l)_{AB_j}\, dt dl \tag{3.39}
$$

By substitution of:

$$
\int_0^t P_c(t,l)_{AB_j}\, dt = \frac{V_c}{4\left(D_A+D_B\right)\pi l}\left(1-erf\left(\frac{l}{\sqrt{4\left(D_A+D_B\right)t}}\right)\right) \tag{3.40}
$$

The previous equation becomes:

$$k_a B_0 t = \frac{1}{\tau_r}\int_0^{+\infty} \frac{4\pi l^2 B_0 N_A V_c}{4(D_A + D_B)\pi l}\left(1 - erf\left(\frac{l}{\sqrt{4(D_A + D_B)t}}\right)\right)dl$$

$$\frac{k_a \tau_r}{4N_A V_c} = \int_0^{+\infty} \frac{l}{4(D_A + D_B)t}\left(1 - erf\left(\frac{l}{\sqrt{4(D_A + D_B)t}}\right)\right)dl$$

$$\frac{k_a \tau_r}{4N_A V_c} = \int_0^{+\infty} \frac{l}{\sqrt{4(D_A + D_B)t}}\left(1 - erf\left(\frac{l}{\sqrt{4(D_A + D_B)t}}\right)\right)d\left(\frac{l}{\sqrt{4(D_A + D_B)t}}\right)$$

$$\text{let } \frac{l}{\sqrt{4(D_A + D_B)t}} = x \text{, then}$$

$$\frac{k_a \tau_r}{4N_A V_c} = \int_0^{+\infty} x\left(1 - erf\left(x\right)\right)dx \tag{3.41}$$

Surprisingly, there is a numerical solution to the integration: $\int_0^{+\infty} x\left(1 - erf\left(x\right)\right)dx = \frac{1}{4}$, which reduces the equation to an elegant form:

$$\frac{k_a \tau_r}{V_c N_A} = 1 \tag{3.42}$$

This states a simple relationship between the association rate constant k_a with the parameters V_c we hypothesized and used in the theory. Here, a dimension analysis was made:

Parameter	Unit (SI)
V_c	m^3
τ	s
k_a	$M^{-1}s^{-1} = mol^{-1}m^3s^{-1}$
N_A	mol^{-1}

$$\frac{[k_a][\tau_r]}{[V_c][N_A]} = \frac{[mol^{-1}m^3s^{-1}][s]}{[m^3][mol^{-1}]} = 1$$

Perfecto! Hence, the hypothetical parameters in the equation can be replaced by the experimentally measurable rate constant k_a:

$$\frac{V_c}{\tau_r} = \frac{k_a}{N_A} \tag{3.43}$$

Hence, the final form of reaction probability between 2 molecules writing with macroscopic parameter is:

$$P_{ij}(t, l_{ij}) = 1 - \exp\left(-\frac{k_a}{4(D_A + D_B)\pi N_A l_{ij}} \left(1 - erf\left(\frac{l_{ij}}{\sqrt{4(D_A + D_B)t}} \right) \right) \right) \tag{3.44}$$

3.7 Dissociation reactions - relating microscopic and macroscopic behaviour

Dissociation reactions can take the following forms:

$$A \xrightarrow{k_d} B$$
$$A \xrightarrow{k_d} B_1 + B_2$$
$$A \xrightarrow{k_d} B_1 + B_2 + B_3 + \ldots$$

As the reagent involves only one molecule, they can be described by one simple deterministic rate equation:

$$\frac{dA(t)}{dt} = -k_d A(t) \tag{3.45}$$

With solution:

$$A(t) = A_0 \exp(-k_d t) \tag{3.46}$$

Since the dissociation reaction is independent of any interactions with other molecules, the reaction probability for each molecule as a function of time is:

$$P_r(t) = 1 - \exp(-k_d t) \tag{3.47}$$

3.8 Summary

In summary, the theory bridges the macroscopic measurable parameters with microscopic parameters needed for the Monte Carlo method. First, the motion of molecules was modelled. The macroscopic diffusion coefficient was related to the microscopic random-walk motion of each molecule. The direction and length of each random step is defined in the spherical coordinate:

$$\theta = rand,\ rand \in [0, 2\pi)$$
$$\phi = \arccos(1 - rand),\ rand \in [0, 2]$$

$$r = F^{-1}(rand),\ rand \in [0,1)$$
$$\text{where } F(r) = erf\left(\frac{r}{\sqrt{4D\Delta t}}\right) - \frac{r}{\sqrt{\pi D\Delta t}}\exp\left(-\frac{r^2}{4D\Delta t}\right)$$

where *rand* is a random number in the specified range, *erf* is the error function, D is the macroscopic diffusion coefficient, and Δt is the simulation time step duration. This model ensures that the simulated diffusion coefficient always agree with the input diffusion coefficient regardless of the Δt chosen for the simulation.

Second, the macroscopic reaction rate constants for association and dissociation reactions are related to the microscopic reaction probability of each molecule. The probability of an association reaction between the two molecules depends on their initial separation (l_0),

diffusion constant (D_A, D_B), the time duration allowed (Δt) and the association reaction rate constant (k_a):

$$P_a = 1 - \exp\left(- \frac{k_f \left(1 - erf\left(\frac{l_0}{\sqrt{4(D_A + D_B)\Delta t}} \right) \right)}{4(D_A + D_B)\pi N_A l_0} \right)$$

The probability of a dissociation reaction depends only on the time duration allowed (Δt) and the association reaction rate constant (k_d):

$$P_d = 1 - \exp\left(-k_r \Delta t\right)$$

Chapter 4

MBS: Monte Carlo Biochemical Reaction Simulator

4.1 Introduction

The MBS package was developed using DEV-CPP in standard C++ and compiled using GCC3.4 (and above). It consisted of a program for simulating biomolecular reactions and a program for visualizing the results. For the simulation of reaction networks, the simulation program was controlled from command line by three setup scripts describing the reaction, the geometry and the experiment time course, respectively. The purpose of the scripts was to avoid modification and recompilation of the source code for different experiments.

4.2 Implementation

The simulation uses equal time step durations to simulate reactions. In each time step, all molecules were randomly moved according to Section 3.3.2 and 3.3.3. Molecules were selected randomly to ensure no molecules react statistically earlier than others to avoid biasing their probability of reaction. For each selected molecule, the type of reactions it can undergo was chosen in random order as well to avoid biasing toward certain types of reactions. Furthermore, if the chosen reaction was an association reaction, reaction partners were chosen at random to again avoid biasing their probability of reaction. The locations of each molecule and the concentration of each type of molecule were recorded in data files for each time step. For the visualization of the results, the program used OpenGL to play back a 3D movie of the molecular movement and reaction. The molecule concentrations over time were stored in a separate data file, which could be imported by standard data analysis software such as Excel or Origin Labs to analyze the reaction kinetics. The following sections will discuss the implementations in detail.

4.2.1 Program architecture

The software package is written with many modules for easy debugging and future development. Each module handles a set of unique functions:

physParam – all the physical parameters are defined here.

randomNumber – generates all the random numbers needed in this program, an essential for Monte Carlo simulation.

Mersenne Twister random number generator – true double precision random number generator seeding randomNumber.

inverse3DGaussian – a custom numerical function, reading pre-tabulated numerical table from file.

dataFetcher – I/O accessing data structure and handles all data I/O exceptions.

geometry – defines the geometric space where molecules can move, it performs checks to determine if all molecules are moving within their regions.

reaction – handles all the reactions (association and dissociation) as well as the order of reactions for all molecules.

diffusion – handles the random walking motion of all molecules, while calling geometry to ensure that all molecules remain in their defined regions.

experiment – provides the time course of the simulation. Molecules can be added or moved at any time step, physical parameters such as temperature can also be changed with respect to time.

main – the main program coordinating the simulation. It is responsible for file I/O and handles all the exceptions.

display – the main program for reading and displaying the output movie file using openGL.

The relationship between these modules is shown in Figure 4-1.

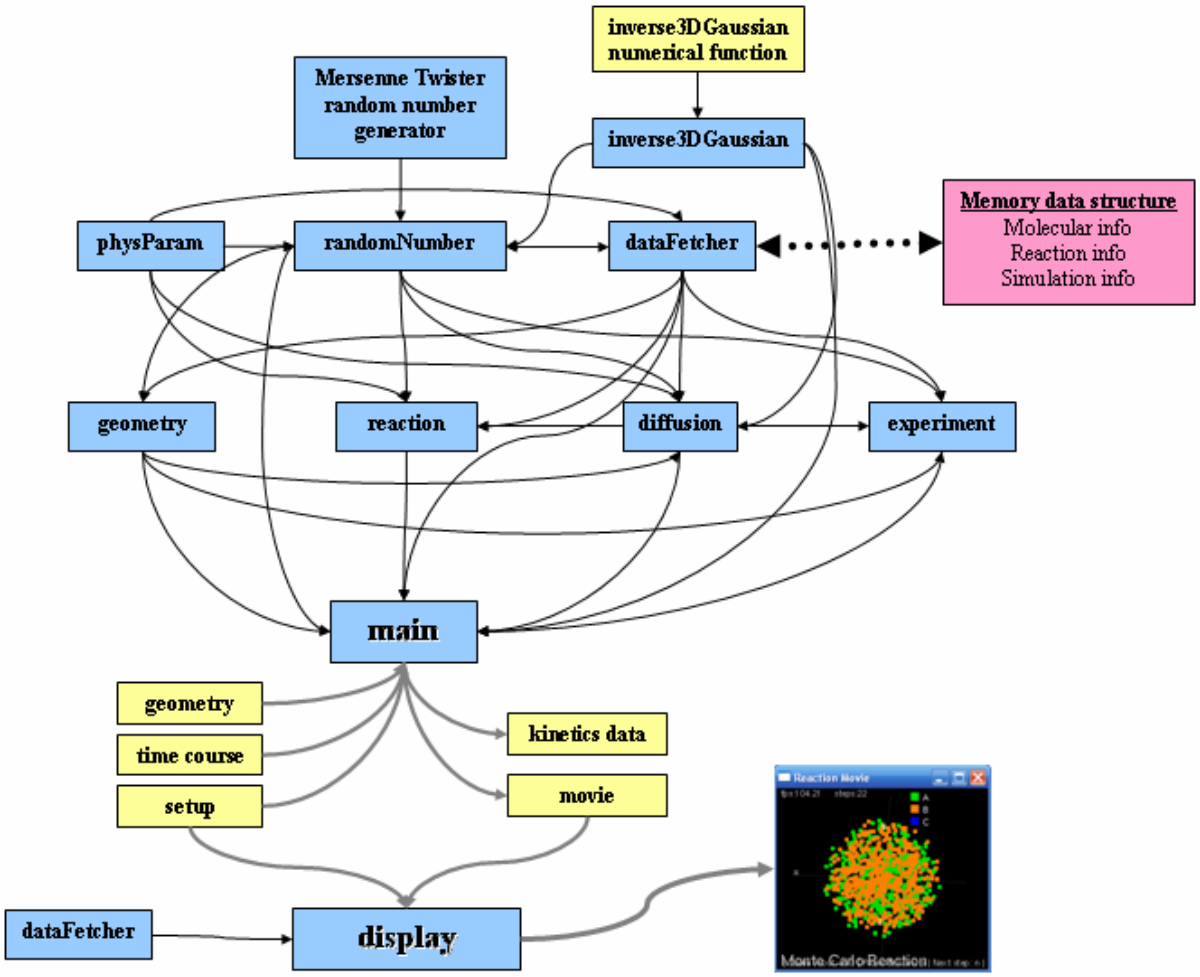

Figure 4-1: Software architecture. Calling between classes, files and data structures. Solid black: program internal calling between classes. Solid grey: file I/O. Dotted black: memory data structure access.

4.2.2 Data structure

For optimal speed performance, a custom memory-based data structure was used. The data structure includes the following parameters:

- Number of molecular species
- Maximal number of molecules per species
- Active number of molecules per species
- For each molecule:
  - Name
  - Active flag of current step
  - Active flag of previous step
  - Location (x, y, z) at current step
  - Location (x, y, z) at previous step
  - Mass
  - Size
  - Diffusion coefficient
  - Region it can travel within
- Number of association reactions
- Number of dissociation reactions
- For each association reaction:
  - Reagent 1
  - Reagent 2
  - Product
  - Association rate constant
- For each dissociation reaction:
  - Reagent
  - Number of product
  - Array of product ID
  - Dissociation rate constant
- Association reaction hash table
- Dissociation reaction hash table

The molecule information of both the current time step and the previous time step are used as required by reaction handling. (See Section 4.2.5) The program requires the total number molecules to be allocated in arrays prior to the simulation such that there is no need to dynamically allocate and destroy molecules during the simulation, for performance optimization. The allocated number of molecules for each species must be enough to cover for all situations. Since each molecule occupies ~50byte in memory only, it is efficient to assume a much higher safe upper bound population for each molecule species as most simulations have a total population around 10000 (~500kB of memory) molecules only. Therefore, whether a molecule is currently participating a reaction or merely a "ghost" depends on the active flag. When the flag becomes true, the molecule is materialized in the simulation that participates in diffusion and reaction. When a reaction occurs and the molecule is turned into something else, the active flag of the reagent molecule becomes "false", and the product becomes "true". The program will never update or consider molecules marked "false" on active flag, hence creating no drawback on performance.

The association reaction hash table is a look up table: given the two reagent molecule IDs, the product molecule ID is the value in the hash table. In addition, the association reaction hash table must be able to handle situation where two reagent molecules could result in more than one product, such as:

$$A + B \xrightarrow{k_{a1}} C$$
$$A + B \xrightarrow{k_{a2}} D$$

For instance, if there are 5 reactions and the molecules are identified by numbers, then:

```
0 + 1 → 2
2 + 3 → 0
```

2 + 4 → 1

2 + 4 → 5

3 + 3 → 4

would produce the following 3 dimensional association reaction hash table:

layer 1: Number of reactions for each reagent pair

	0	1	2	3	4	5
0	0	**1**	0	0	0	0
1	**1**	0	0	0	0	0
2	0	0	0	**1**	**2**	0
3	0	0	**1**	0	0	0
4	0	0	**2**	0	0	0
5	0	0	0	0	0	0

layer 2: First reagent molecule ID for each pair

	0	1	2	3	4	5
0	-1	**2**	-1	-1	-1	-1
1	**2**	-1	-1	-1	-1	-1
2	-1	-1	-1	**0**	**1**	-1
3	-1	-1	**0**	**4**	-1	-1
4	-1	-1	**1**	-1	-1	-1
5	-1	-1	-1	-1	-1	-1

layer 3: Second reagent molecule ID for each pair

	0	1	2	3	4	5
0	-1	-1	-1	-1	-1	-1
1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	**5**	-1

```
3|   -1   -1   -1   -1   -1   -1
4|   -1   -1   5    -1   -1   -1
5|   -1   -1   -1   -1   -1   -1
```

The first layer of the hash table indicates the number of reactions for each pair of reagents. In this case, there are two reactions possible for reagent molecule ID 2 and 4. The second and third layers are the product molecule ID for each pair. A "-1" indicate that those pairs do not react. The dissociation reaction table was constructed similarly.

4.2.3 Handling diffusion in restricted regions

In unrestricted space, diffusion is handled by equations of motion in Section 3.3.2 and 3.3.3. However, most biological systems have boundaries such as the cell wall or the nuclear membrane. They create spatial regions where molecules cannot escape. Spatial regions are created here by defining simple geometric shapes (Table 4-1):

Table 4-1: Region types and definition parameters.

Region shape	***Definition parameters***
Box	Centre (x, y, z) and dimension (width, length, height)
Box shell	Centre (x, y, z), dimension of the inner box ($width_1$, $length_1$, $height_1$) and of the outer box ($width_2$, $length_2$, $height_2$)
Sphere	Centre (x, y, z) and radius (r)
Sphere shell	Centre (x, y, z), inner radius (r_1) and outer radius (r_2)
Cylinder	Centre (x, y, z), radius (r) and length (l)
Cylinder shell	Centre (x, y, z), inner radius (r_1), outer radius (r_2) and length (l)

Each region is defined by a unique region ID. And each molecular species is marked with region IDs they are allowed to move in. Because the regions are well defined simple geometric shapes, it is easy to check whether a molecule is within a region. Before the location of a molecule is assigned in the next time step by diffusion, a candidate location is created first. If the candidate location is inside the assigned region, it is assigned as the real molecule location. However, if the candidate location drifts outside the assigned region, a new candidate location is generated until the new location is within the assigned region.

In simulations, it should be kept in mind that the time step duration should not be too large such that the diffusion random walk step length is never larger than the dimensions of the region. In that case, it would be extremely computationally expensive to diffuse molecules in a small region as many trials must be performed. In situations where the random walk step length is much greater than the dimension of the region, it can usually be assumed that the molecule distribution within that region is homogeneous. Hence, a smaller substitutive diffusion coefficient can be used to reproduce the same concentration homogeneity within the regions in return for higher computational efficiency.

4.2.4 Handling single reactions

To determine whether a reaction will occur, their reaction probability is calculated according to Section 3.6 (for association reactions) and Section 3.7 (for dissociation reactions). This reaction probability (value between 0 and 1) is then compared with a random number taking value between 0 and 1 to determine whether the reaction occurs or not. If the random

number is less than or equal to the reaction probability, a reaction will occur. Otherwise, the reaction does not occur. The location of the product molecule C is on the line segment joining the two reagent molecules A and B where the ratio of distance AC to CB was proportional to their average random walk step length.

If an association reaction does occur in the previous time step, the reagent molecules are removed from both the previous and current time step, whereas the product molecule is created only in the current time step. Avoiding creating and destroying molecules from the same molecule data structure allows the program to know the molecules that have attempted reaction and the ones that are newly created. To ensure the correct overall kinetics, these molecules should not participate in reactions for a second time in order.

4.2.5 Handling multiple reactions

Suppose a molecule A can participate in multiple association and dissociation reactions:

$$A + B \xrightarrow{k_{a1}} C$$
$$A + D \xrightarrow{k_{a2}} E$$
$$A \xrightarrow{k_d} F + G$$
$$A \xrightarrow{k_d} H + I$$

Since our Monte Carlo simulation uses discrete time steps, it must decide for each molecule A, whether it will react and if so, which reaction it will undergo. For each molecule A at every time step, the four possible reactions are shuffled in random orders. They are then tried one by one to see whether a reaction would occur. If a reaction does occur, the rest of the reactions are not tried. Each reaction is handled as described in the previous section (Section 4.2.4). All reactions involving molecule A are tried instead of just randomly

picking one reaction is because the rate of reaction is additive when more possible reactions involving molecule A are possible. If only one reaction is chosen, it is equivalent to dividing the probability (and rate) of each reaction by the number of total number of reactions. Hence, all reactions must be tried within one time step to recreate the same rate for each reaction.

4.2.6 Random number generation

In C++, the default random number generator generates a random integer between 0 and 32767. A double precision number generated from this random integer has a maximal resolution of 32767 bins. Hence, for deciding whether a reaction would occur (Section 4.2.4), the smallest value of the random number is 0 and the second smallest value would be $1/32767=3.05\times10^{-5}$. Hence, if a reaction has a theoretical probability of less than 3.05×10^{-5}, the simulated probability would the same because 0 random number value would always allow the reaction to occur, which has a constant probability of 3.05×10^{-5}. Due to this error, all simulated kinetics curves association reactions with rate constants smaller than $10^{-5}\ M^{-1}s^{-1}$ converge.

This problem was solved by introducing a true double precision random number generator using the Mersenne Twister algorithm, extending the lower bound of the random number to $1/1.8\times10^{308} \approx 5.5\times10^{-309}$, well beyond any physically realistic of rate constants.

4.2.7 Optimizations

The following optimizations were made to improve the performance of the program:

Complexity reduction

The reaction probability between every pair of reagent molecules are calculated for each time step. This is the most computational expensive calculation as the complexity scales with the number of reactions ($N_{reaction}$) and the square of number of reagent molecules ($N_{reagent}$):

$$O(\Omega) = N_{reaction} \times N_{reagent}^2$$

For a normal simulation with $10^5 – 10^6$ molecules, this would be $10^{10} – 10^{12}$ calculations per time step, too slow for a standard PC.

To reduce the complexity, we assumed that two molecules can never react if they are beyond their diffusion limit. The diffusion limit distance is chosen as two times of the diffusion profile width. When two molecules are beyond this distance apart, their reaction probabilities are practically 0. Doing so, the majority of the molecules do not need to participate in the time consuming computation of reaction probability. The complexity of the computation is effectively reduced to:

$$O(\Omega) = N_{reaction} \times N_{reagent}$$

Although the complexity of the most expensive function call is reduced, one still needs to calculate the distance between every pair of reagent molecules to determine which pairs are selected for reaction determination. We expect development in the future to optimize this.

Data structure optimization

Static hash tables are used instead of dynamic data structures to save the computational time from excessive memory allocation/delocation, I/O and searches. Static tables are good

choices for this application because the reaction information does not change, and the size of molecule information does not fluctuate significantly.

Minor function optimizations

- Functions are optimized by reducing the amount of unnecessary math operations. For instance, in a function comparing the distance between two points with a fixed value could be speed up 5 fold by not using the square root function. Instead, comparisons were made between the squares of the two distances.

- Conditions yielding straight results were put at the beginning of the function such that no computations are performed when the conditions are met.

- All the symmetrical matrix calculations were performed on only half of the matrix to save computational time.

- Frequently accessed functions are written as inline functions to save the function calling time.

- The code was compiled using best optimization method offered in C++: "-fexpensive-optimizations -O3".

4.3 Using MBS

4.3.1 Running MBS

There are five files necessary for running the MBS simulation:

- **MBS.exe** – the executable
- **inverse3DGaussian.function** – numerical function tabulation
- **setup** – text file containing molecular and reaction information
- **experiment** – text file scripting the time course of the experiment
- **geometry** – text file specifying the geometric constraints of the reaction

MBS.exe must be called from command line. There are three ways to call the MBS.exe:

```
MBS.exe setup experiment geometry -m movie
MBS.exe setup experiment geometry -d data
MBS.exe setup experiment geometry -m movie -d data
```

The setup, experiment and geometry file names must be in order following the MBS.exe. There are three options to output the movie file, the data file, or both. If the input information contains error, or any of the script files contains error, the program will exit with an error message showing the potential source of error. If the parameters are correct, the program will continue executing. One should expect the following on the command window screen (Figure 4-2 and Figure 4-3):

```
Trying to understand what you want to do...understood!

        output movie to: movie
        output data to: data.txt


_____________________________________________________________________

                            Initialization
_____________________________________________________________________


Initializing random number generator:
        normal C++ rand()...DONE
        Mersenne Twister rand gen...DONE

Initializing numerical function:
        Loading inverse3DGaussian.function...DONE

Initializing simulation parameters:
        Opening setup file : setup.txt...DONE
        Parsing setup info and allocating memory:
                reading and allocating total # of species.......DONE
                reading and allocating each species.............DONE
                allocating 'new' time step.....................DONE
                loading reaction information...................DONE
                loading environment and simulation parameters...DONE
Simulation parameters initialized successfully!

Initializing geometry of regions:
        Opening setup file : geo.txt...DONE
        Parsing geometry region parameter table................DONE

Initializing experiment time course:
        Opening setup file : exp.txt...DONE
        Parsing experiment time course table.....................DONE
species\ts      0       1500    2000    3000
        0       500     0       0       0
        1       500     0       0       0
        2       0       -1000   -1000   -1000
        T       0       0       0       0
        pH      0       0       0       0
regionID  type  center  other parameters
0  sph  (0.0e+000 0.0e+000 0.0e+000) (1.0e-006)
```

Figure 4-2: Running MBS screenshot. Initialization and parsing setup, experiment and geometry scripts.

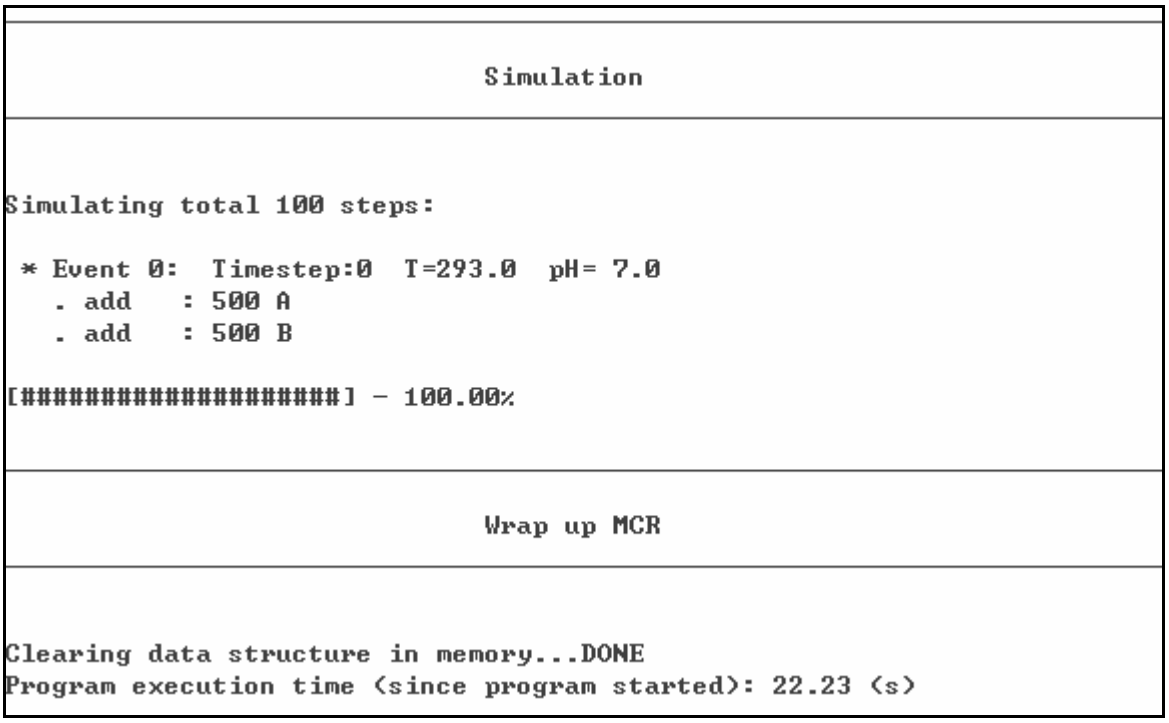

Figure 4-3: Running MBS screenshot. Simulation and total runtime.

To ensure the program has parsed all the information in the script file correctly, the experiment time course and molecule information are displayed for double-checking. (Figure 4-2) As the program runs, the number of molecules actually added and removed as specified by the experiment time course is displayed as well as a progress bar. When the program finishes execution, it displays the total runtime. (Figure 4-3)

4.3.2 Viewing molecule movie

There are three files required for running the MBSmovie simulation:

- **MBSmovie.exe** – the executable
- **setup** – text file containing molecular and reaction information

- **movie** – the output movie file

MBSmovie.exe must also be called from command line with the following command:

```
MBSmovie.exe setup movie
```

An OpenGL window should be displayed as shown in Figure 4-4. The window displays the colour coding for each molecule and the current time step index. The movie is played back in loops. The movie can be paused or resumed by pressing "p". The motion of the mouse controls the viewing angle of the whole reaction. Zooming in and out is by pressing "a" and "z", respectively.

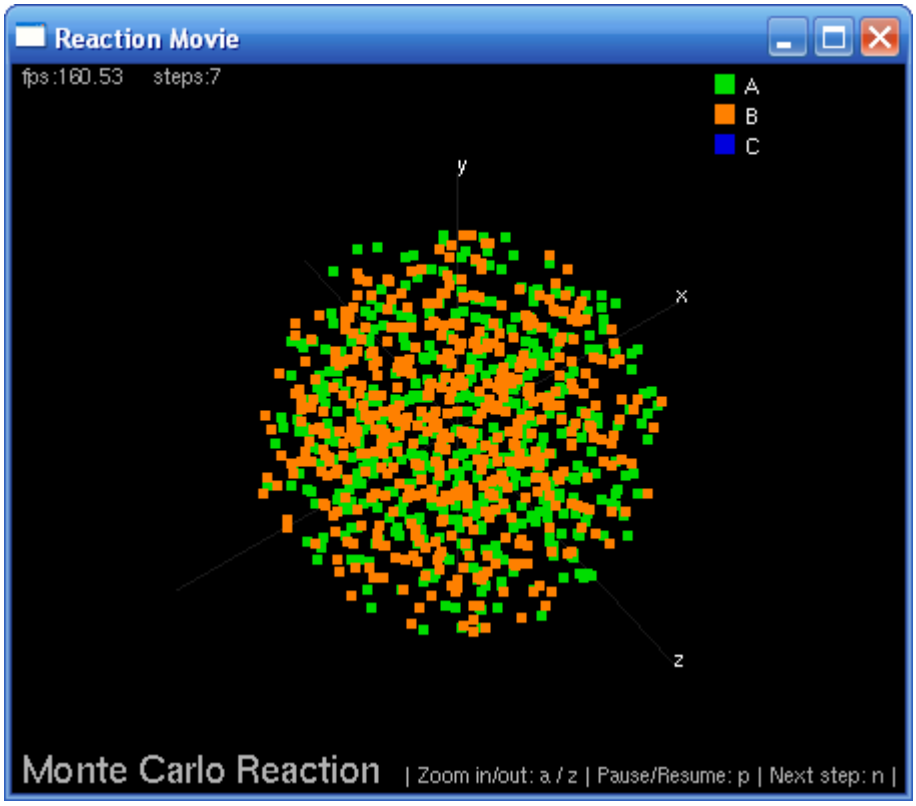

Figure 4-4: MBSmovie displaying a reaction in a spherical region. The fps and current step index are displayed on the left upper corner. The colour map for different molecules is displayed on the upper right corner. On-screen instructions are displayed at the bottom of the screen.

4.3.3 Input scripts

Three aspects of the simulation: reaction, time course, and geometry are divided into three input scripts. Three scripts are used instead of combining them into one because in most situations, we want to tune the simulation by modifying information in only one of the aspects and keep the rest unchanged. The information included in each script is summarized in Table 4-2.

Table 4-2: Details included in the setup scripts for each simulation

Molecule and reaction	Geometry	Time course

▪ Molecules • Diffusion coefficient • Size • Mass ▪ Reaction • Chemical formula • Rate constants ▪ Setups • pH • Temperature • Viscosity • Time step duration • Total simulation duration	▪ A combination of: • Sphere • Box • Cylinder • Spherical shell • Box shell • Cylindrical shell	▪ Adding or moving molecules at arbitrary time step ▪ Changing environment such as temperature or pH at arbitrary step

Setup script

The setup script provides detailed molecule information such as their diffusion constant, maximal expected population and the region they belong to. In addition, reactions as well as their rate constants are defined in the setup (Figure 4-5):

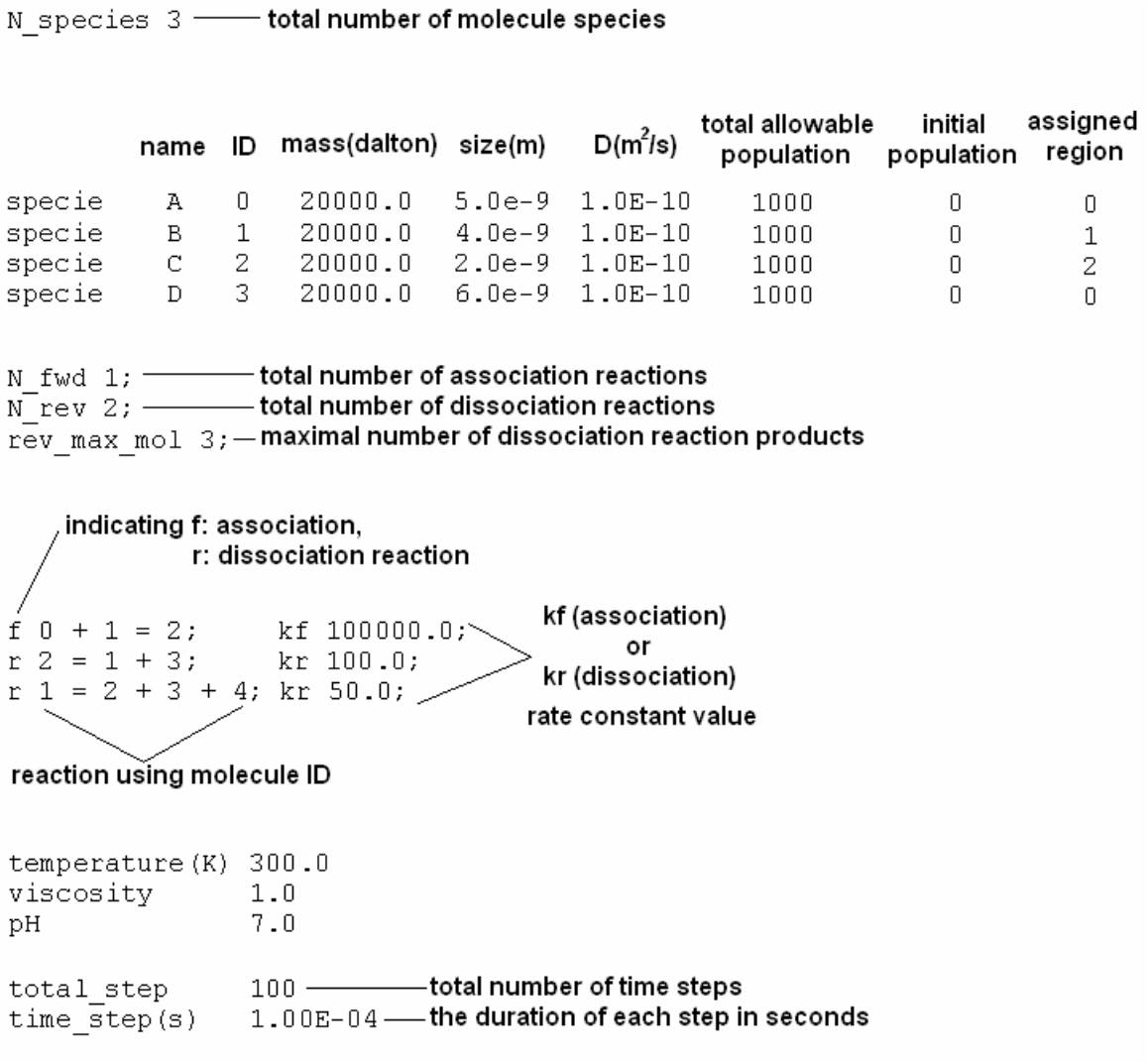

Figure 4-5: Format of the setup script.

Experiment script

In many cases, we would like to study how the biomolecular system responds to external stimuli. Hence, the program allows addition and removal of molecules at any time during the simulation. This experiment information is contained in the experiment script. (Figure 4-6)

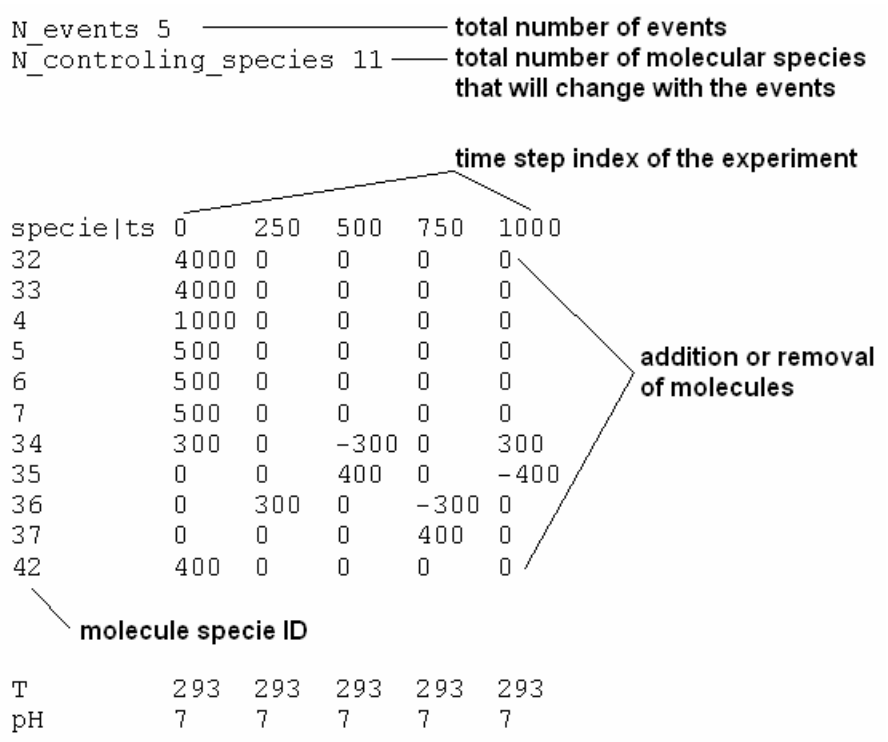

Figure 4-6: Format of the experiment script.

Geometry script

This script specifies the spatial regions where different molecules can move within. The 6 types of geometric regions are defined in Table 4-1, and the setup of the script is shown in Figure 4-7:

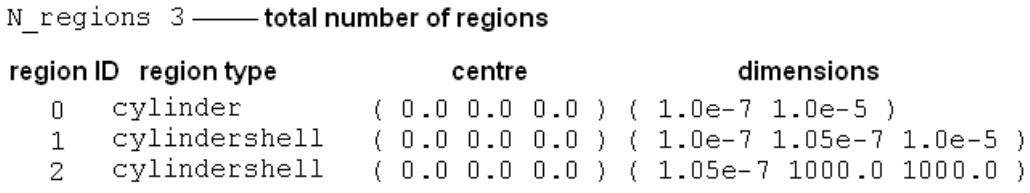

Figure 4-7: Format of the geometry script.

4.3.4 Output file format

Kinetics data

The kinetics data composes of the total population of each molecular species for every time step in a text file (Figure 4-8):

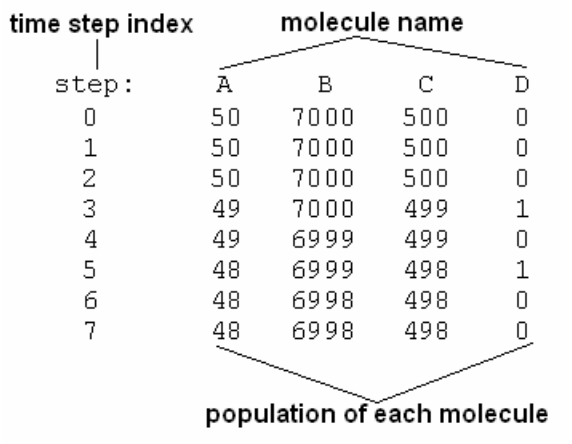

Figure 4-8: Format of kinetics data output file.

Reaction movie

The reaction movie file records the location of every molecule throughout the reaction. The reaction movie can be used to reconstruct the kinetics data, as the active population of each specie is known. The format of the reaction movie file is shown below (Figure 4-9):

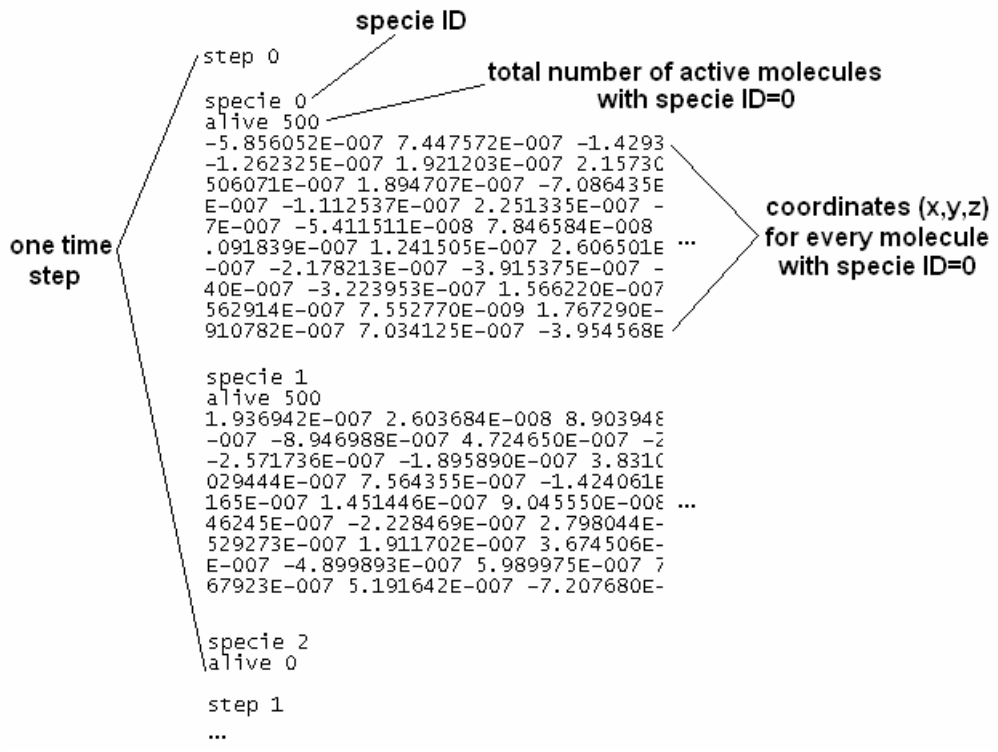

Figure 4-9: Format of reaction movie output file.

Chapter 5

Model systems

The content of this chapter was modified from the peer-review journal paper:

I. T. S. Li and K. Truong, "A computation tool for Monte Carlo simulations of biochemical reactions modeled on physical principles," *Bioinformatics (submitted)*, 2007.

5.1 Overview

The following model systems are studied in this chapter:

- **Diffusion from a single source** – verification of our Monte Carlo diffusion method with deterministic solution. The advantage and accuracy of the method demonstrated by comparison with another method used in literature.
- **Simple reaction kinetics** – verification of the association, dissociation reaction kinetics by comparing to deterministic kinetics. The equilibrium points of reversible reactions were also verified with deterministic kinetics.
- **The predator-prey model** – demonstrating the spatial and temporal molecular fluctuation greatly influence the overall reaction kinetics.
- **The genetic oscillator** – genetic oscillator simulated in prokaryotic cells are compared with that in eukaryotic cell to show how the localization of DNA in the eukaryotic nucleus changes the behaviour of the genetic oscillator.
- **Ca^{2+} wave along a cylindrical compartment** – showing how the geometric distribution of membrane channels changes the fast kinetics of Ca^{2+} wave propagation.
- **Design of a chemical memory unit with synthetic protein network** – a protein network was designed and simulated to show that it is capable of memorizing a chemical state. The behaviour is modeled analogous to that of a D-latch commonly used in electrical circuits.

5.2 Diffusion verification

5.2.1 Method

To test our diffusion model, 10^4 non-interacting molecules were initially placed at the location of one point and allowed to diffuse over time. Then, the molecule distribution in our simulated diffusion was verified to be correct by comparing the result to the deterministic solution. Next, the diffusion process was verified to be independent to the Δt by modeling identical diffusion processes using D=10^{-10} m^2s^{-1} under different Δt ranging from 10^{-5} s to 10^{-3} s.

5.2.2 Results and discussions

Our diffusion model was accurate in describing the physical process of diffusion at the different time step durations (Δt). Since the diffusion process is independent of Δt, a difference in Δt should ideally not affect the diffusion kinetics, but instead only change the temporal resolution. To test this condition, 10^4 molecules were placed at a single point and diffused with various Δt. With the same initial condition and total simulation duration, the population distributions (Figure 5-1A) were identical using different Δt in the simulations. Furthermore, the population distribution coincided with the deterministic distribution described by the macroscopic diffusion equation:

$$\frac{\partial \phi(\vec{r},t)}{\partial t} = D\nabla^2 \phi(\vec{r},t)$$

where D was the diffusion coefficient, q was the density distribution as a function of r (the vector from the centre of diffusion to the point of interest) and t (the total duration of diffusion). Hence, our diffusion model produced the correct spatial profile of molecules, which provided a solid foundation for accurately assessing reaction kinetics.

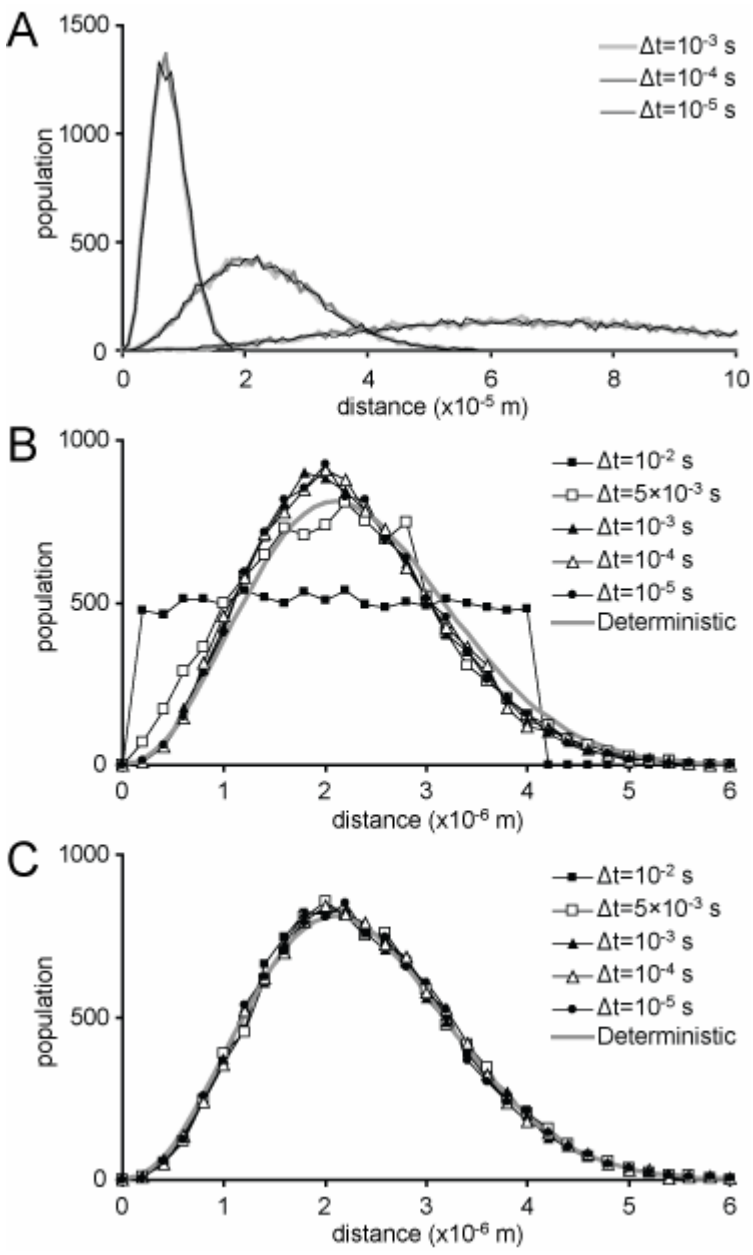

Figure 5-1: A. The population distribution of 10^4 molecules was diffused from a single point as a function of their distances from the origin of diffusion. The three distinctive humps are the population distribution of different total time durations of 10 ms, 100 ms and 1000 ms. The three shades are simulations under different time step duration Δt=10^{-3} s (lighter grey), 10^{-4} s (darker grey) and 10^{-5} s (black). The coincidence of the three shaded curves shows that diffusion is independent to the Δt. The same diffusion coefficient D=10^{-10} m^2/s is used. The population distribution of B. uniform step size distribution model and C. our diffusion model show good agreement with deterministic solution for our model and disagreement for the uniform distribution model. Both simulations have a total duration of 0.1 s. The deterministic solution is indicated as the thick grey line. The population distributions using 5

time step durations Δt=10^{-2} s (solid square), 5×10^{-3} s (hollow square), 10^{-3} s (solid triangle), 10^{-4} s (hollow triangle), and 10^{-5} s (solid circle) were compared with the deterministic solution.

In contrast, a uniform step size model used in recent literature was not accurate in describing the physical process of diffusion. In this model, the random walk step size used a uniform distribution from 0 to a maximal value depending on D and Δt. A uniform step size model was compared to our diffusion model by simulating total time duration of 10^{-1} s under different Δt's (10^{-1} s, 10^{-2} s, 10^{-3} s and 10^{-4} s). For different Δt, the uniform step size model produced population distribution that does not coincide with each other (Figure 5-1B, C). The difference was especially evident if Δt was large compared to the total simulation duration. Since reactions in Monte Carlo simulations are handled by finding reaction probabilities that are highly dependent on distances between reagent molecules, an inaccurate spatial distribution of molecules caused by uniform step size model will yield an inaccurately simulation of reaction kinetics.

5.3 Basic reaction kinetics verification

5.3.1 Method

To test simple reaction kinetics, association and dissociation reactions were simulated using parameters that resemble conditions inside cells: biomolecules were homogeneously distributed in a spherical volume with a radius of 1 μm; the concentration of biomolecules was varied in the range of 100 nM to 10 μM; association rate k_a was varied from 10^4 to 10^6 $M^{-1}s^{-1}$; dissociation rate k_d was varied from 1 to 100 s^{-1}. Lastly, the resulting kinetics curves were compared with the deterministic solutions.

5.3.2 Results and discussions

At the different time step durations (Δt), our reaction model accurately simulated simple reaction kinetics as the kinetics curves of the simulation coincided with the deterministic curves. For the association reaction defined by $A + B \xrightarrow{k_a} C$, an initial concentration of 198 μM (or 500 molecules/cell) for both A and B was created in a spherical cell with a radius of 1 μm. The association rate constants k_a in the physiological range of 10^3 – 10^8 $M^{-1}s^{-1}$ were tested. Deterministically, the kinetics of this reaction was described by the differential equation:

$$\frac{d[C]}{dt} = k_a[A][B]$$

Since A and B had the same initial concentration A_0, the equation was simplified:

$$\frac{d[C]}{dt} = k_a\left(A_0 - [C]\right)^2$$

The deterministic curve coincided with kinetic curve from the simulation in the k_a range of 10^3 – 10^6 $M^{-1}s^{-1}$ (Figure 5-2A). Furthermore, within this range, the simulation curves with different Δt coincided with the deterministic curves (data not shown). Notice that at k_a above 10^7 $M^{-1}s^{-1}$, the deterministic rate of reaction became faster than the simulated rate as the reaction became diffusion limited. This physical phenomenon occurred in extreme situations when reactions happen faster than molecules could diffuse into areas depleted of reagent molecules. This lowered the effective local concentration of the reagents and therefore the speed of reaction becomes slower. This phenomenon of diffusion limited reactions was

easily captured by our model, however using a deterministic model, it cannot without additional modifications.

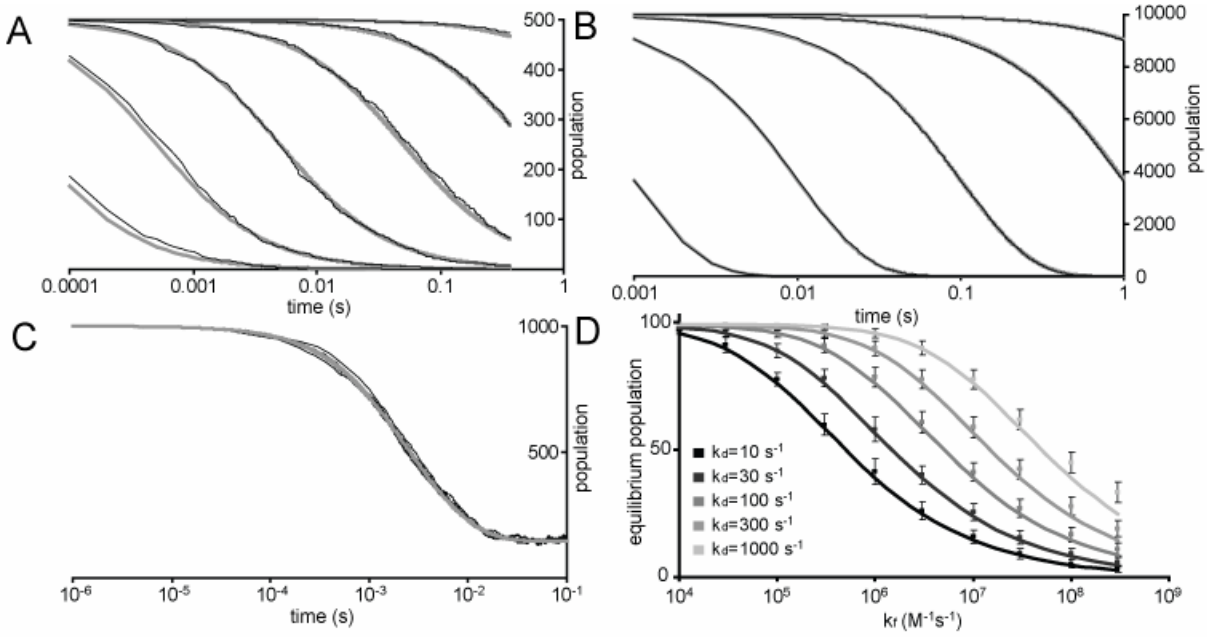

Figure 5-2: A. Population of reagent molecule as a function of time plotted in log scale for association reactions with various rate constant k_a ranging from 10^3 to 10^8 $M^{-1}s^{-1}$. B. Population of reagent molecule as a function of time plotted in log scale for dissociation reaction with various rate constant k_d ranging from 10^{-1} to 10^3 s^{-1}. C. Reversible reaction kinetics showing population of reagent molecule as a function of time plotted in log scale for association reactions with various simulation time step durations Δt=10^{-6} s to 10^{-3}s. In all the above figures, the simulated kinetics curve (thin black) is compared to the deterministic kinetics curve (thick grey). The Δt is 0.0001s and the total duration is 0.5 second. D. equilibrium constants in k_a-k_d space matches what is predicted by the deterministic kinetics equations.

For the dissociation reaction defined by $A \xrightarrow{k_d} B + C + \ldots$, the deterministic rate equation was $\frac{d[A]}{dt} = -k_d[A]$ with a solution: $[A] = [A_0]\exp(-k_d t)$. Our simulation kinetic curve coincided with the deterministic curves for the entire range of k_d (Figure 5-2B). Again, this agreement was independent of Δt of the simulation (data not shown).

Reversible reactions were studied by combining the association reaction and the dissociation reaction:

$$A + B \underset{k_d}{\overset{k_a}{\rightleftharpoons}} C$$

At different Δt, the simulated kinetics curves coincided with the deterministic curves and reached the same equilibrium points (Figure 5-2C). To verify this for a broader range of k_a and k_d, 50 reactions with an array of k_a and k_d values were simulated to equilibrium point. The simulated equilibrium concentration of molecule C is compared to what is expected from deterministic kinetics. The agreement was well established until reaching k_a above 10^8 $M^{-1}s^{-1}$ which was an expected result from diffusion limited reactions (Figure 5-2D).

5.4 The predator-prey model

5.4.1 Method

The oscillator was simulated using the following biomolecular reactions with association and dissociation parameters that ensured an oscillation would occur:

$$A + X \xrightarrow{k_{a1}} 2X \qquad k_{a1}=2\times10^5\ M^{-1}s^{-1}$$

$$X + Y \xrightarrow{k_{a2}} 2Y \qquad k_{a2}= 5\times10^6\ M^{-1}s^{-1},$$

$$Y \xrightarrow{k_d} D \qquad k_{d3}=200\ s^{-1}$$

The total simulation duration was 1 s with a Δt of 0.1 ms. The diffusion coefficients D of all molecular species were varied in the range of 10^{-10} to 10^{-12} m^2s^{-1}.

5.4.2 Results and discussions

Spatial heterogeneity of a molecular specie arising from molecular fluctuations created local areas of reaction kinetics that differ from the population. The predator-prey model was used to construct a simple spatial and temporal biomolecular oscillator. In this model, there was a constant source of molecule A that can be converted into molecule X by X at the rate of k_{a1}. And similarly, X can be converted into molecule Y by Y at the rate of k_{a2}, which naturally decayed into D at the rate of k_d. The below equation described the biomolecular network:

$$A + X \xrightarrow{k_{a1}} 2X$$
$$X + Y \xrightarrow{k_{a2}} 2Y$$
$$Y \xrightarrow{k_d} D$$

Deterministically, this was described as follows:

$$\frac{dX(t)}{dt} = k_{a1}A(t)X(t) - k_{a2}X(t)Y(t)$$
$$\frac{dY(t)}{dt} = k_{a2}X(t)Y(t) - k_dY(t)$$

where *A(t)*, *X(t)*, and *Y(t)* were the concentration over time of A, X, and Y, respectively. The solution to this system of differential equations was a stable oscillation in the concentration of A, X and Y with a fixed amplitude (Figure 4-3A, B). However, using our simulation showed that the concentration oscillation amplitude was chaotic (Figure 4-3C,D). While the reaction started with a spatial homogeneity of molecular species, as reactions occurred, it created local areas of spatial heterogeneity (Figure 4-3E). The differences in local concentrations caused local differences in the reaction kinetics, resulting in off-phase concentration oscillations. Since the overall oscillation was a superposition of the local

oscillations, off-phased local oscillations caused an overall oscillation that was less coherent and had varying amplitude peaks.

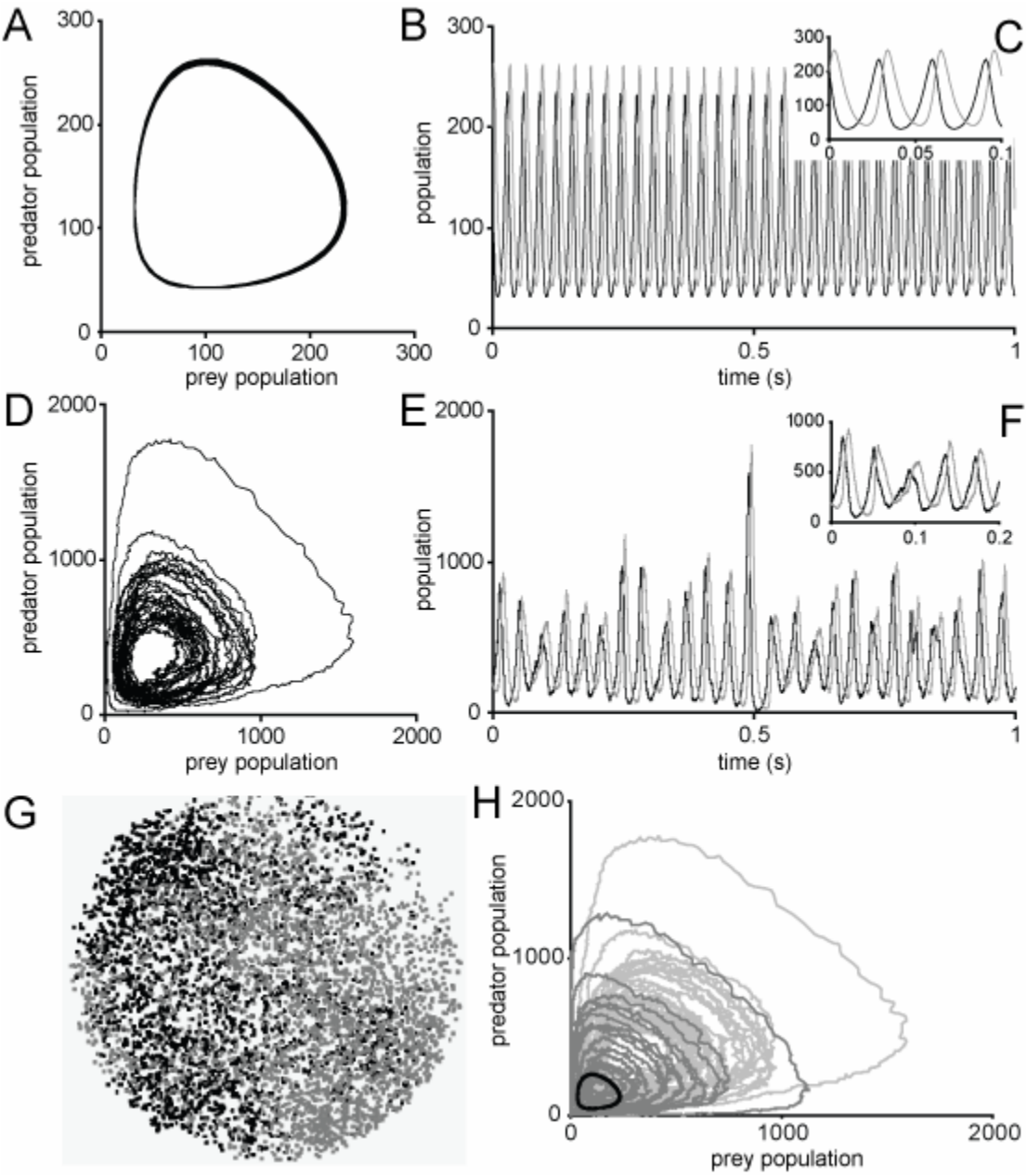

Figure 5-3: Deterministic model solutions of A. phase space predator population vs. prey population, B. predator (grey) and prey (black) population over time, C. zoomed in view of the kinetics curves. Monte Carlo simulation solution of D. phase space predator population vs. prey population, E. predator (grey) and prey (black) population over time, F. zoomed in view of the kinetics curves. G. spatial heterogeneity

of the reaction, showing large spatial fluctuation. Black dots: prey, grey dots: predators. F. demonstration of diffusion coefficient changes the frequency and amplitude of the simulation in phase space. Lower diffusion coefficient (lighter grey) results in curves with greater amplitudes while higher diffusion coefficient (darker grey) shifts the curves towards the bottom left corner. The deterministic solution is shown in black.

The effect of spatial heterogeneity was reduced by a larger diffusion coefficient (such as 10^{-10} m^2s^{-1}) which made the molecular species more homogenous over the course of the simulation. Using this diffusion coefficient, the peak amplitude of population oscillation was less varied, resembling the oscillatory amplitude of the deterministic solution (Figure 4-3F). In addition, because the amplitude was smaller, the oscillation frequency was higher as it took less time to traverse the predator-prey population phase space where the path length was shorter. In contrast, a small diffusion coefficient (10^{-12} m^2s^{-1}) made the molecules less mobile and hence kept the molecular species more localized. Therefore, a large local concentration can be achieved, which lowered the oscillation frequency (data not shown). Hence, the diffusion coefficient played an important role in determining biomolecular kinetics.

5.5 Genetic oscillator

5.5.1 Method

Genetic circuits for both prokaryotic and eukaryotic cell were constructed using the following parameters: inhibitor-DNA complex dissociation constant: 10^{-7} M, mRNA synthesis rate: 500 s^{-1}, protein synthesis rate: 500 s^{-1}, mRNA degradation rate: 50 s^{-1}, protein degradation rate: 50 s^{-1}, copy number of plasmid DNA: 50

The total simulation duration was 3 s with Δt of 1 ms. Both the prokaryotic and eukaryotic cell had a spherical volume with a radius of 2 μm. In the case of the eukaryotic cell, it had a spherical nucleus with a radius of 0.75 μm, which contained the DNA.

5.5.2 Results and discussions

The spatial localization of molecules into cellular compartments changed the amplitude or phase of the genetic oscillator. To study the effect of molecular localization in compartments, a prokaryotic genetic oscillator was compared to a eukaryotic genetic oscillator. In our prokaryotic oscillator, mRNA was transcribed and translated into proteins in the cytoplasm. These translated protein then regulated gene expression by interacting with genetic material in the cytoplasm. Conversely, in our eukaryotic oscillator, mRNA was transcribed inside the nucleus and then transported outside the nucleus where mRNA was translated into proteins. For these translated protein to regulate gene expression, they were transported back into the nucleus. Since transcription and translation were relatively slow reactions, low reaction rate constants were used. Thus, any biomolecular species had sufficient time to reach spatial homogeneity within its own compartment before the reaction kinetics changes significantly.

The genetic network was constructed to produce oscillating concentration of three different species of proteins each with a phase shift as previously described (Figure 5-4C). [23] The network was composed of three repressor proteins (tetR, lacI and λcI) forming a cyclic negative feedback loop - each inhibiting the expression of one other repressor (See Figure 5-4C). The repressor protein and mRNAs were constantly degraded by intracellular proteases and RNase. For instance, if the expression of tetR was inhibited, tetR protein

concentration decreased. This decrease released the repression of λcI, causing the concentration of λcI to increase. This cycle propagated and repeated to create an oscillation in the concentration of each of the repressor proteins with the same delay from each other. Transcription, translation and degradation (of both protein and mRNA) were modeled in our simulation as simple first order reactions as the detailed stages in these processes were too complex to be modeled precisely. For comparison purposes, the synthesis and degradation rates of RNA and protein were the same for both prokaryotic and eukaryotic cells. The binding between the regulatory regions of the plasmid DNA and the three repressor proteins were modeled as reversible reactions with the same dissociation constants in both systems.

In the simulation of the prokaryotic case, the three repressor proteins oscillated with $2\pi/3$ phase difference as expected due to the symmetry of the system (Figure 5-4A), while the oscillation characteristics of the eukaryotic were dependent on the transport of biomolecules across the nuclear membrane. As expected in the prokaryotic oscillator, the molecular fluctuation caused the variations in both the amplitude and phase of the oscillations similar to the predator-prey model. In the eukaryotic oscillator, the periodic oscillation pattern was no longer present in the simulation when using the same set of initial conditions (Figure 5-4B). This was due to insufficient transport of mRNA and proteins across the nuclear membrane. The statistical fluctuation of the nuclear import and export rate in combination with the phase delay caused by the nuclear transport destroyed the synchrony between the protein expression and gene repression. This caused the concentration oscillation of the three repressor proteins to be off from the $2\pi/3$ phase and hence the chaotic fluctuation (Figure 5-4B). When the rate of nuclear transport was sufficiently high, the oscillation returned

because the phase delay became negligible between the cytoplasmic and the nuclear concentration.

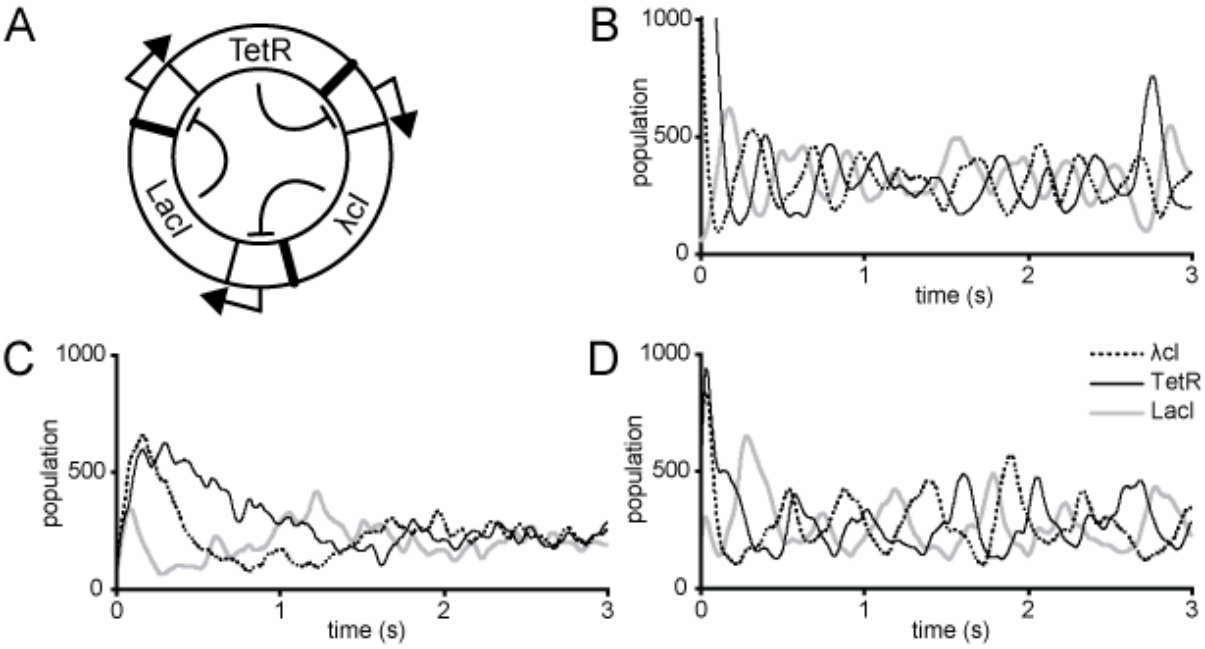

Figure 5-4: A, the plasmid schematic of a genetic oscillator. Regions with arrows were repressor binding regions of the plasmid DNA. Regions marked with λcI, TetR and LacI are the genes regulated by their respective repressor binding regions. B, the population changes of the three repressor proteins in the prokaryotic model. C, D, the population oscillations of the three repressor proteins in the eukaryotic model with C, low nuclear membrane transport rates of 5×10^{3} s^{-1} and D, high nuclear membrane transport rate of 5×10^{5} s^{-1}. Dotted lines represented λcI; solid black, TetR; solid grey, LacI.

5.6 Ca2+ wave

5.6.1 Method

A cylindrical compartment with a radius of 0.1 μm and a length of 10 μm was constructed. The membrane of the compartment was embedded with Ca^{2+}-dependent calcium channels (CDCC) at a range of concentrations. These CDCCs bind to outside Ca^{2+} which opens the

channel, releasing Ca^{2+} inside the compartment. The flux of Ca^{2+} depends on the concentration gradient across the membrane, similar to the IP_3 receptor proteins on the surface of endoplasmic reticulum (ER). In our simulation, the Ca^{2+} concentration inside the compartment was much higher than outside, hence, when Ca^{2+} induced the initial release of Ca^{2+}, these Ca^{2+} ions would bind to more CDCCs and release more Ca^{2+} in a positive feedback loop. The Ca^{2+} concentration inside the compartment was similar to the physiological values inside ER ~500 μM.

5.6.2 Results and discussions

Our simulations showed that the kinetics of Ca^{2+} wave propagation depended on the density and geometric arrangement of the Ca^{2+} channels. Through the highly regulated events of intracellular Ca^{2+} homeostasis, often seen as a Ca^{2+} wave, Ca^{2+} regulates numerous physiological cellular phenomena including development, differentiation and apoptosis. When triggered by other secondary messengers such as IP_3 (inositol-1,4,5-triphosphate), Ca^{2+} is released from the ER to the cytoplasm by the channels such as the IP_3 receptor. To simulate a Ca^{2+} wave, we modeled the Ca^{2+}-induced-Ca^{2+}-release mechanism, where the membrane Ca^{2+} channels were opened when they bound to Ca^{2+} from the outside. A cylinder representing the ER was lined with Ca^{2+} channels and filled with Ca^{2+}. Hence, when a dose of Ca^{2+} was added to one end of the cylinder, the edge of the Ca^{2+} diffusion triggered the adjacent channel to release Ca^{2+}, reinforcing the Ca^{2+} wave propagation in that direction (Figure 5-5).

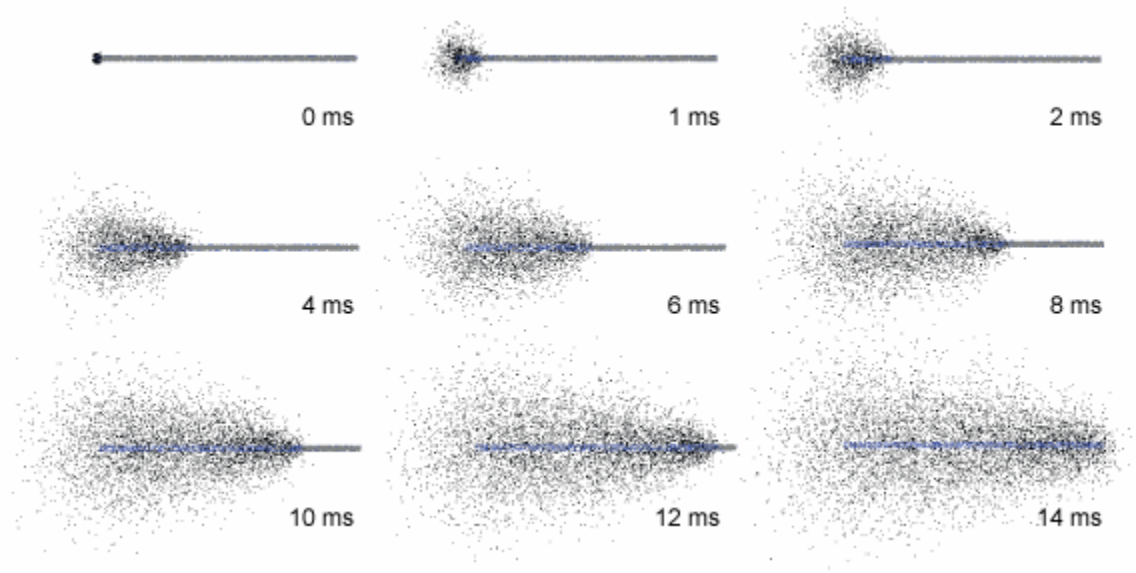

Figure 5-5: The spatial propagation of the Ca^{2+} ions in time. Top-left corner shows the triggering event at t=0 ms. The ion propagation to the right can be seen moving much faster than diffusion, hence the ion distribution profile is elongated in the horizontal direction.

The surface density of Ca^{2+} channels as well as their Ca^{2+} transport rate affected the speed of wave propagation. In these simulations, the Ca^{2+} channel release rate was in the same order as Ca^{2+} diffusion. If the binding and channel opening events were slower than Ca^{2+} diffusion, the speed of Ca^{2+} wave was dominated by diffusion, resulting in a wave propagation less guided by the cylinder (Figure 5-6C). Conversely, if the binding and channeling opening were faster than Ca^{2+} diffusion, the wave propagation speed was faster than diffusion (Figure 5-6A, B). The Ca^{2+} channel density also played a very vital role to the propagation of the Ca^{2+}. In the case with 1000 channels on the membrane, there were sufficient channels on the membrane to maintain a high Ca^{2+} release rate of ~ 420 ions/ms until 14 ms (Figure 5-6D, E). At 14 ms, the Ca^{2+} release rate decreased to 0 as the propagation reached the end of the cylindrical compartment. Multiple runs of the same simulation showed that the propagation rate and the duration of propagations are consistent (Figure 5-6D, E). In contrast, in the case

with only 100 channels on the membrane, the propagation along the cylindrical chamber took over twice as long as in the case of 1000 membrane channels for an average of 32ms (Figure 5-6F). Additionally, the Ca^{2+} release rate is no longer uniform throughout the propagation (Figure 5-6G) and is inconsistent among simulations with same setup parameters (Figure 5-6F, G). This is due to decreased uniformity of membrane channels along the axial direction as the number of membrane channels is lowered. In a situation with non-uniform distribution of membrane channels, the regions with higher channel density has a faster propagation rate and vice versa. For instance, the simulation shown as black curves in Figure 5-6F, G had a faster Ca2+ release and consequently faster propagation rate till 15ms after the propagation started. The propagation then slowed down due to sparser channels in the middle section of the cylinder.

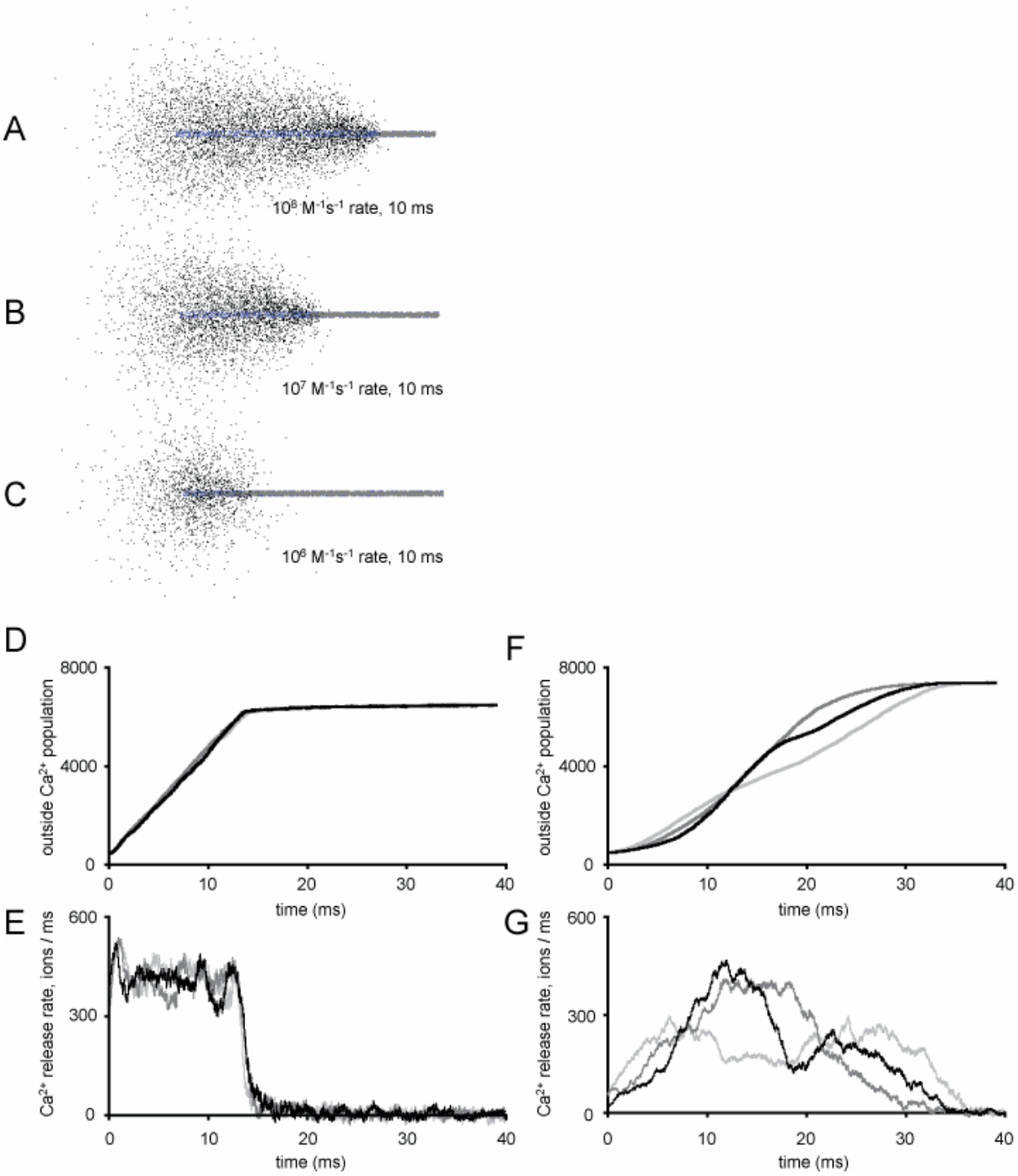

Figure 5-6: Graphical representation of the Ca^{2+} distribution after 10 ms for experiments with different channel binding/opening rate: A, 10^8 $M^{-1}s^{-1}$, B, 10^7 $M^{-1}s^{-1}$, and C, 10^6 $M^{-1}s^{-1}$. D, the Ca2+ population outside the compartment over time for 1000 membrane channels. E, Ca2+ release rate over time for 1000 membrane channels F, the Ca2+ population outside the compartment over time for 100 membrane channels. G, Ca2+ release rate over time for 100 membrane channels. Black, dark grey and light grey lines in each figure (D, E, F, G) are three simulations using the same setup parameters.

5.7 A protein reaction network that implements a D-latch memory element

5.7.1 Method

The association rate constant of maltose to N_{data} (MBP*) and N_{clock} (MBP) were 10^8 and 10^9 $M^{-1}s^{-1}$, respectively. The dissociation rate constant of maltose from both MBP* and MBP were 10^3 s^{-1}. All protein-protein interaction had an association rate constant of 10^6 $M^{-1}s^{-1}$ and a dissociation rate constant of 100 s^{-1}. The association and dissociation rate constants of Q and P to $N_{output2}$ and $N_{output1}$ were 10^8 $M^{-1}s^{-1}$ and 5×10^3 s^{-1}. The catalytic rate of Q and P creation was 5×10^4 s^{-1}, while the degradation rate of Q and P was 1.6×10^4 s^{-1}.

5.7.2 Results and discussions

Using synthetic proteins composed of WW domains, maltose binding protein (MBP), calmodulin, adenylate cyclase, guanylate cyclase, cAMP binding domain from protein kinase A (PKA), and cGMP binding domain from protein kinase G (PKG), a biomolecular reaction network was simulated that behaved similar to a digital memory element (i.e D-latch), showing that a network of proteins can remember state (Figure 5-7A). D-latches are simple digital memory units that synchronize the setting of a data bit (value that can be 0 or 1) to a clock signal (also can be 0 or 1). When the clock is 1, the D-latch maintains the previously set value; when the clock is 0, the D-latch output is changed to the input D. D-latches are often incorporated in digital systems to remember the state of a machine and the machine

responds specifically to inputs depending on its state. Similarly, cells have the notion of a state as it responds differently to the same stimuli such as in stem cell differentiation. We showed that a network of proteins with particular interactions can create a D-latch and therefore a protein network can be used to encode states. Since a D-latch is made fundamentally by switches in a particular configuration, proteins engineered to perform switching can theoretically create similar memory units. A synthetic binding or catalytic protein can be engineered to have two regulatory sites, where each site serves as an input signal and the binding or catalytic activity of the protein serves as the output signal. Furthermore, this protein can be engineered such that as long as one site is occupied (representing a 1 input), the binding activity is inhibited (representing a 0 output). This switching behaviour is known as a NOR logic gate. From digital logic theory, we can create a D-latch using 4 NOR logic gates (Figure 5-7A).

Engineering protein logic gate is inspired by how natural proteins perform switching and logic functions. For example, the protein calmodulin (CaM) acts like a Ca^{2+} regulated switch as it undergoes conformational change upon Ca^{2+} binding and is then able to bind to other proteins. [97, 98] Artificial protein switches has also been constructed in the past by combining two functional distinct proteins such as the β-lactamase (BLA) and maltose-binding protein (MBP), where MBP controls the activity of BLA when triggered by maltose sugar. [99-101] Similarly, protein logic gates can be implemented by combining two switching mechanisms arranged in the fashion to provide the desirable logic function.

As signals of our synthetic protein network, we designate 0 as the absence or inactivity and 1 as the presence or activity of any biomolecular specie or protein. The input signal D and Clock are chosen as small molecules because their absence or presence could be experimentally controlled more easily. A synthetic protein can be engineering using a binding protein with two regulatory sites where each site serves as an input signal and the binding activity of the protein serves as the output signal. Furthermore, this protein can be engineered such that as long as one site is occupied, the binding activity is turned off. This switching behaviour is known as a NOR logic gate. From digital logic theory, we can create a D-latch using 4 NOR logic gates (Figure 5-7A). The nature of the protein logic gate is decided by the mode of interaction among the binding sites and the protein itself. For instance, a protein NAND gate requires both binding partners to be present for the protein to be inactive, where as for a protein NOR gate, as long as one binding site is occupied, the protein's activity is turned off. This implies that constructing certain protein logic gates such as the NOR gate would be considerably easier than a NAND gate. Hence, the D-latch was modified to a clock-inverted version, thereby replacing the 4 NAND gates in the circuit with 4 NOR gates

Specifically, Ca^{2+} and maltose were chosen as inputs to Data and Clock, respectively. Their concentration can be controlled *in vitro* by directly addition and dialysis. Choosing signaling molecules with different size and charge allows independent control of the input signaling molecules by selective dialysis. The output of the two NOR gates, N_{data} and N_{clock}, were the binding activity of the WW_1 and WW_2 domain, respectively (Figure 5-7). The synthetic protein N_{data} can be engineered by fusing WW_1 domain with both CaM and MBP such that

when either Ca^{2+} or maltose is present, the activity of N_{data} is inhibited (Figure 5-7). Similarly, N_{clock} was created using WBP_1 (a binding partner for WW_1) and MBP to receive input that inhibits the function of WW_2 (Figure 5-7B). Since N_{data} had two inhibitory sites responding to input signals whereas N_{clock} only had one, when both Ca^{2+} and maltose signals were present, N_{data} was inhibited much higher than N_{clock}. To correct for the imbalance, a lower affinity MBP* was used on N_{data}. The output of memory module $N_{output1}$ and $N_{output2}$ were the activities of adenylate cyclase (AC) and guanylate cyclase (GC) that produced cAMP and cGMP, which are constantly degraded by background phosophodiesterases. The synthetic protein $N_{output1}$ and $N_{output2}$ were also designed with cGMP and cAMP binding domains from protein kinase G (PKG) and protein kinase A (PKA) such that when bind to cGMP and cAMP, they would inhibit the catalytic activity of $N_{output1}$ and $N_{output2}$, respectively (Figure 5-7). Thus, in any steady equilibrium state, either $N_{output2}$ or $N_{output1}$ is active but not both. Furthermore, $N_{output1}$ and $N_{output2}$ were connected to the logic module by the activity of N_{data} and N_{clock}, respectively. When N_{data} or N_{clock} was active, its WW_1 or WW_2 domain bound and inhibited $N_{output1}$ or $N_{output1}$, respectively. This can be described by the following biomolecular reactions (Figure 5-7):

$$N_{data} + N_{output1} \rightarrow inactive\_N_{output1}$$
$$N_{clock} + N_{output2} \rightarrow inactive\_N_{output2}$$
$$P + N_{output1} \rightarrow inactive\_N_{output1}$$
$$Q + N_{output2} \rightarrow inactive\_N_{output2}$$
$$N_{output2} \rightarrow N_{output2} + P$$
$$N_{output1} \rightarrow N_{output1} + Q$$
$$P \rightarrow degraded$$
$$Q \rightarrow degraded$$

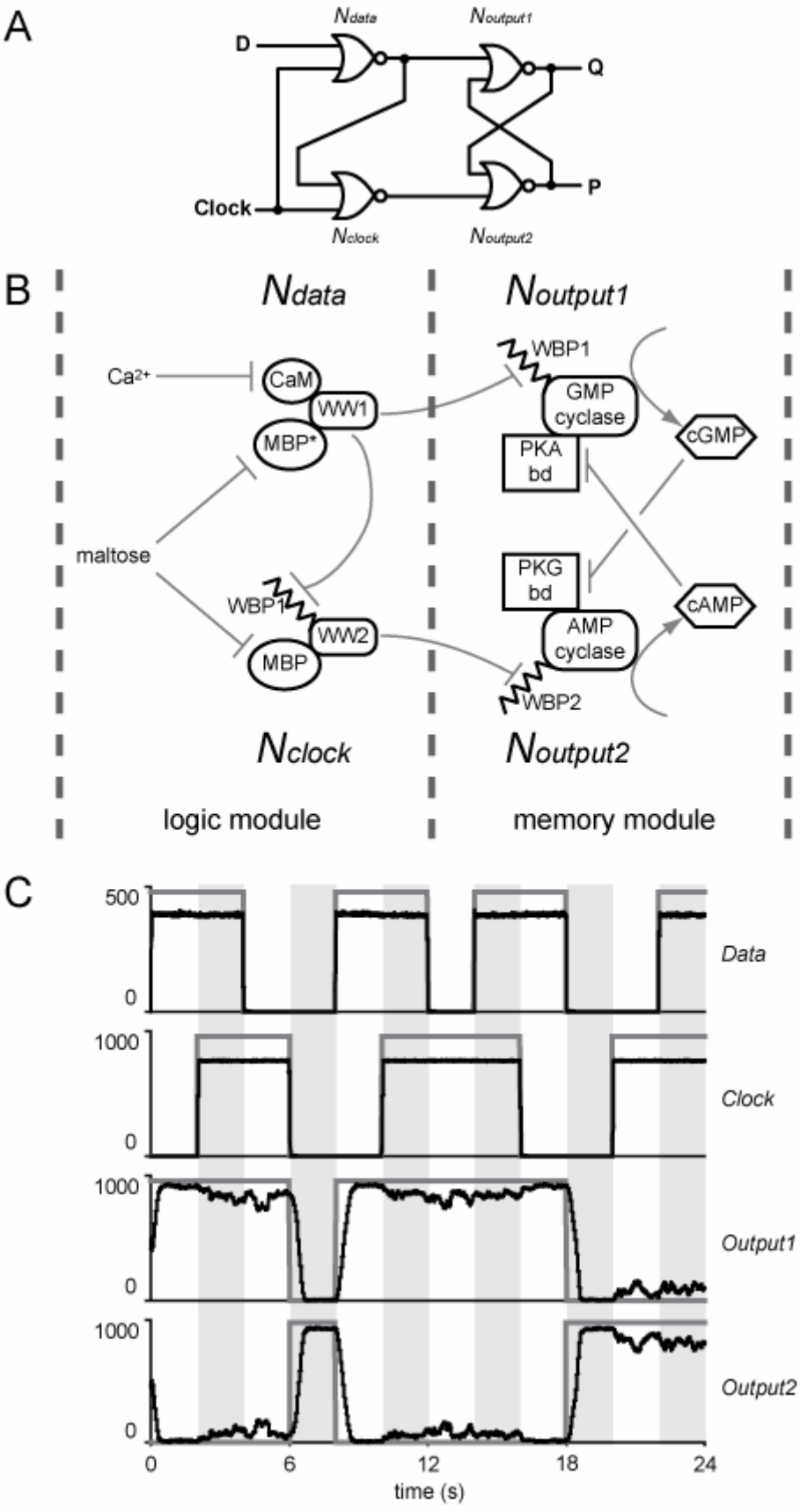
A
Ndata
Noutput1
D
Q
P
Clock
Nclock
Noutput2
B
Ndata
Noutput1
Ca2+
CaM
WW1
MBP*
WBP1
GMP cyclase
PKA bd
cGMP
maltose
PKG bd
cAMP
WBP1
WW2
MBP
AMP cyclase
WBP2
Nclock
Noutput2
logic module
memory module
C
500
0
Data
1000
0
Clock
1000
0
Output1
1000
0
Output2
0
6
12
18
24
time (s)

Figure 5-7: A. A clock inverted D-latch in electronics representation, the four NOR gates are names N_{data}, N_{clock}, $N_{output1}$, and $N_{output2}$. B. Protein circuit implementation of the D-latch. C. the timing diagram of electronic D-latch in comparison with protein D-latch. Grey lines indicate the response expected for digital circuit, black lines are the response from the protein circuit.

To achieve significant bi-state behavior, the population of cGMP and cAMP generated in either state should greatly exceed the population of $N_{output2}$ and $N_{output1}$ when a state is maintained. The large population of cGMP and cAMP ensures the stability of the memory state. On one hand, we need to maintain the state of this memory module, on the other hand, we must be able to switch it to the opposite state when needed. For instance, if we want to switch from $N_{output1}$ active to $N_{output2}$ active, we must ensure that when the switching signal is triggered, active population of $N_{output1}$ must be brought down below the population of $N_{output2}$. This requires the dissociation constant between the N_{data} and $N_{output1}$ be sufficiently low such that N_{data} is able to deactivate more $N_{output1}$ than cGMP could inhibit $N_{output2}$.

We optimized the protein circuit by tweaking the kinetic parameters. It was found that when the binding on/off rates in protein-protein (N_{data}, N_{clock}, $N_{output2}$, $N_{output1}$) interactions is much lower than in protein-signaling-molecule (Ca^{2+}, maltose, cGMP, cAMP) interactions, the system shows higher signal-to-noise ratio. We reasoned this is because when the interaction between signaling molecules with proteins are weak, a large population of signaling molecules is required, which would have a relatively smaller statistical variation. Consequently, the binding affinity between protein and small molecules would be much smaller than between protein and protein. This ensures the stability of the circuit as the effect of signaling molecule fluctuations is buffered by the tight interactions between proteins. Due to the symmetry of the memory module, the binding affinity between $N_{output1}$, $N_{output2}$

and their substrates should be as close as possible. A large difference in binding affinity results in a preferential (biased) state in the memory module, which makes state flipping difficult and the system is prone to flipping state by itself. Lastly, the stoichiometry of the 4 proteins in the network must be precisely 2:1:1:1 (N_{data}:N_{clock}:$N_{output1}$:$N_{output2}$) to ensure the correct signal levels in the circuit. Insufficient input level would make it difficult and sometimes impossible to flip the state of the memory module due to the activities of residual protein species.

After fine tuning the kinetics parameters and stoichiometry, our protein network functions like a D-latch (Figure 5-7C, D). The concentration of the maltose and Ca^{2+} were controlled, while the changes in concentration of $N_{output2}$ and $N_{output1}$ were tracked over time. When the maltose concentration (representing the Clock) was set to low, the adenylate cyclase activity followed the concentration of Ca^{2+} (representing D). When the maltose concentration was high, both the adenylate cyclase and guanylate cyclase activity maintained previous levels despite the changes in Ca^{2+} concentration. The unevenness of the output was due to molecular fluctuations as there were only 500 molecules of $N_{output1}$ and $N_{output2}$ in the simulation.

Chapter 6

Conclusion

Systems biology and synthetic biology demand quantitative methods to study biomolecular systems and networks. A survey in these fields was conducted showing the importance and necessity to create an accurate simulation tool to study biomolecular networks. Here, we described the creation of a computational tool using a Monte Carlo approach for simulating the spatial and temporal kinetics of biomolecular reaction networks within a cell. Since our models were based on physical principles, the tool accurately produced diffusion and reaction constants across a range of time step durations. Simulations on a predator-prey model demonstrated the phenomenon of spatial heterogeneity and its effect on the frequency and amplitude of the oscillation. Subsequent simulations on prokaryotic and eukaryotic genetic oscillators demonstrated transport of proteins and mRNA across a nuclear compartment disturbs the oscillation. In fast reactions such as Ca^{2+} waves, density and geometric arrangement of the Ca^{2+} channels affect the speed of Ca^{2+} wave propagation. Lastly, using known activities of protein domains, we constructed a synthetic protein reaction network that functioned like a digital memory element, showing that biomolecular networks are capable of remember states and changing its behaviour depending state. Together this work demonstrates the unique insights that can be discovered by considering the subtle effects that can be created by the spatial and temporal kinetics of biomolecular reaction networks. Future applications of the computational tool include designing synthetic networks or modeling larger existing biological networks.

6.1 Ongoing projects and future work

Currently, we are exploring various ways to construct protein circuits analogous to electronic circuit components such as logic gates, latches and flip-flops. These will provide the necessary modules for constructing more complex interacting networks. Methods to interconnect various modules using signaling molecules or interacting protein domains are also being developed.

In addition, we are investigating in the cross-talk between biomolecular networks. Signaling in biological systems is not as specific as in electrical systems, where each wire connects the input to the output. For instance, protein kinases usually have more than one phosphorylation targets due to the non-specific interactions between the kinase and the substrates. As there are many kinases in the system, there are inevitably many cross-talking between different phosphorylation pathways. We want to study how this cross-talking between networks changes the behaviour of each network and the behaviour of the overall system.

Lastly, to make the software package more user-friendly, a graphical user interface (GUI) is being developed to replace manually writing the setup script files. An additional feature would be to plot the kinetics data as the program runs such that adjustments to the reaction parameters can be made earlier, making it more efficient to use.

References

[1] I. T. S. Li and K. Truong, "A computation tool for Monte Carlo simulations of biochemical reactions modeled on physical principles," *Bioinformatics (submitted)*, 2007.

[2] I. T. S. Li, W. Shum, and K. Truong, "160-fold acceleration of the Smith-Waterman algorithm using a field programmable gate array (FPGA)," *BMC Bioinformatics*, vol. 8, pp. 185, 2007.

[3] I. T. S. Li, K. R. Ranjith, and K. Truong, "Sequence reversed peptide from CaMKK binds to calmodulin in reversible Ca2+ -dependent manner," *Biochem Biophys Res Commun*, vol. 352, pp. 932-5, 2007.

[4] I. T. S. Li, E. Pham, and K. Truong, "Protein biosensors based on the principle of fluorescence resonance energy transfer for monitoring cellular dynamics," *Biotechnol Lett*, vol. 28, pp. 1971-82, 2006.

[5] I. T. S. Li and K. Truong, "FRET evidence that an isoform of Caspase-7 binds but does not cleave its substrate," *Biochemical and Biophysical Research Communications (submitted)*, 2007.

[6] E. Pham, J. Chiang, I. T. S. Li, W. Shum, and K. Truong, "A computational tool for designing FRET protein biosensors by rigid-body sampling of their conformational space," *Structure*, vol. 15, pp. 515-23, 2007.

[7] E. Pham, I. T. S. Li, and K. Truong, "Computational Modeling Approaches for Studying of Synthetic Biological Networks," *Current Bioinformatics (submitted)*, 2007.

[8] J. J. Chiang, I. T. S. Li, and K. Truong, "Creation of circularly permutated yellow fluorescent proteins using fluorescence screening and a tandem fusion template," *Biotechnol Lett*, vol. 28, pp. 471-5, 2006.

[9] I. T. S. Li, E. Pham, and K. Truong, *Current approaches for engineering proteins with diverse biological properties*: Landes Bioscience, 2007.

[10] I. T. S. Li and K. Truong, "Monte Carlo simulation of biological reactions with spatial and temporal resolution," presented at 2nd Annual Canadian Student Conference in Biomedical Computing, London, Ontario, Canada, 2007.

[11] I. T. S. Li, W. Shum, and K. Truong, "160-fold acceleration of Smith-Waterman algorithm using field programmable gate array (FPGA)," presented at 2nd Annual Canadian Student Conference in Biomedical Computing, London, Ontario, Canada, 2007.

[12] I. T. S. Li, J. J. Chiang, and K. Truong, "FRET evidence that an isoform of caspase-7 binds but does not cleave its substrate," presented at 28th Annual International IEEE Engineering Conference in Medicine and Biology New York, 2006.

[13] J. J. Chiang, I. T. S. Li, and K. Truong, "The FPMOD: A modeling tool for sampling the conformational space of fusion proteins," presented at 28th Annual International IEEE Engineering Conference in Medicine and Biology, New York, 2006.

[14] B. A. Mello and Y. Tu, "Perfect and near-perfect adaptation in a model of bacterial chemotaxis," *Biophys J*, vol. 84, pp. 2943-56, 2003.

[15] T. M. Yi, Y. Huang, M. I. Simon, and J. Doyle, "Robust perfect adaptation in bacterial chemotaxis through integral feedback control," *Proc Natl Acad Sci U S A*, vol. 97, pp. 4649-53, 2000.

[16] H. H. McAdams and L. Shapiro, "Circuit simulation of genetic networks," *Science*, vol. 269, pp. 650-6, 1995.

[17] R. J. Tanaka, H. Okano, and H. Kimura, "Mathematical description of gene regulatory units," *Biophys J*, vol. 91, pp. 1235-47, 2006.

[18] G. Weng, U. S. Bhalla, and R. Iyengar, "Complexity in biological signaling systems," *Science*, vol. 284, pp. 92-6, 1999.

[19] W. Weber and M. Fussenegger, "Artificial mammalian gene regulation networks-novel approaches for gene therapy and bioengineering," *J Biotechnol*, vol. 98, pp. 161-87, 2002.

[20] C. C. Guet, M. B. Elowitz, W. Hsing, and S. Leibler, "Combinatorial synthesis of genetic networks," *Science*, vol. 296, pp. 1466-70, 2002.

[21] A. Becskei and L. Serrano, "Engineering stability in gene networks by autoregulation," *Nature*, vol. 405, pp. 590-3, 2000.

[22] T. S. Gardner, C. R. Cantor, and J. J. Collins, "Construction of a genetic toggle switch in Escherichia coli," *Nature*, vol. 403, pp. 339-42, 2000.

[23] M. B. Elowitz and S. Leibler, "A synthetic oscillatory network of transcriptional regulators," *Nature*, vol. 403, pp. 335-8, 2000.

[24] D. Bray, "Protein molecules as computational elements in living cells," *Nature*, vol. 376, pp. 307-12, 1995.

[25] R. Unger and J. Moult, "Towards computing with proteins," *Proteins*, vol. 63, pp. 53-64, 2006.

[26] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray, "From molecular to modular cell biology," *Nature*, vol. 402, pp. C47-52, 1999.

[27] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, "Network motifs in the transcriptional regulation network of Escherichia coli," *Nat Genet*, vol. 31, pp. 64-8, 2002.

[28] T. Akutsu, S. Miyano, and S. Kuhara, "Identification of genetic networks from a small number of gene expression patterns under the Boolean network model," *Pac Symp Biocomput*, pp. 17-28, 1999.

[29] M. Chaves, E. D. Sontag, and R. Albert, "Methods of robustness analysis for Boolean models of gene control networks," *Syst Biol (Stevenage)*, vol. 153, pp. 154-67, 2006.

[30] M. Ramachandra, A. Rahman, A. Zou, M. Vaillancourt, J. A. Howe, D. Antelman, B. Sugarman, G. W. Demers, H. Engler, D. Johnson, and P. Shabram, "Re-engineering adenovirus regulatory pathways to enhance oncolytic specificity and efficacy," *Nat Biotechnol*, vol. 19, pp. 1035-41, 2001.

[31] J. A. Zwiebel, "Cancer gene and oncolytic virus therapy," *Semin Oncol*, vol. 28, pp. 336-43, 2001.

[32] C. R. Cho, M. Labow, M. Reinhardt, J. van Oostrum, and M. C. Peitsch, "The application of systems biology to drug discovery," *Curr Opin Chem Biol*, vol. 10, pp. 294-302, 2006.

[33] J. R. Heath, J. F. Stoddart, and R. S. Williams, "More on molecular electronics," *Science*, vol. 303, pp. 1136-7; author reply 1136-7, 2004.

[34] K. Sundaram and T. S. Srinivasan, "Computer simulated modeling of biomolecular systems," *Comput Programs Biomed*, vol. 10, pp. 29-33, 1979.

[35] T. Ideker, T. Galitski, and L. Hood, "A new approach to decoding life: systems biology," *Annu Rev Genomics Hum Genet*, vol. 2, pp. 343-72, 2001.

[36] M. T. Borisuk and J. J. Tyson, "Bifurcation analysis of a model of mitotic control in frog eggs," *J Theor Biol*, vol. 195, pp. 69-85, 1998.

[37] K. C. Chen, A. Csikasz-Nagy, B. Gyorffy, J. Val, B. Novak, and J. J. Tyson, "Kinetic analysis of a molecular model of the budding yeast cell cycle," *Mol Biol Cell*, vol. 11, pp. 369-91, 2000.

[38] J. S. Edwards, R. U. Ibarra, and B. O. Palsson, "In silico predictions of Escherichia coli metabolic capabilities are consistent with experimental data," *Nat Biotechnol*, vol. 19, pp. 125-30, 2001.

[39] M. Morohashi, A. E. Winn, M. T. Borisuk, H. Bolouri, J. Doyle, and H. Kitano, "Robustness as a measure of plausibility in models of biochemical networks," *J Theor Biol*, vol. 216, pp. 19-30, 2002.

[40] W. Salamonsen, K. Y. Mok, P. Kolatkar, and S. Subbiah, "BioJAKE: a tool for the creation, visualization and manipulation of metabolic pathways," *Pac Symp Biocomput*, pp. 392-400, 1999.

[41] T. D. Garvey, P. Lincoln, C. J. Pedersen, D. Martin, and M. Johnson, "BioSPICE: access to the most current computational tools for biologists," *Omics*, vol. 7, pp. 411-20, 2003.

[42] S. P. Kumar and J. C. Feidler, "BioSPICE: a computational infrastructure for integrative biology," *Omics*, vol. 7, pp. 225, 2003.

[43] H. Kitano, A. Funahashi, Y. Matsuoka, and K. Oda, "Using process diagrams for the graphical representation of biological networks," *Nat Biotechnol*, vol. 23, pp. 961-6, 2005.

[44] P. Dhar, T. C. Meng, S. Somani, L. Ye, A. Sairam, M. Chitre, Z. Hao, and K. Sakharkar, "Cellware--a multi-algorithmic software for computational systems biology," *Bioinformatics*, vol. 20, pp. 1319-21, 2004.

[45] P. K. Dhar, T. C. Meng, S. Somani, L. Ye, K. Sakharkar, A. Krishnan, A. B. Ridwan, S. H. Wah, M. Chitre, and Z. Hao, "Grid cellware: the first grid-enabled tool for modelling and simulating cellular processes," *Bioinformatics*, vol. 21, pp. 1284-7, 2005.

[46] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer, "COPASI--a COmplex PAthway SImulator," *Bioinformatics*, vol. 22, pp. 3067-74, 2006.

[47] S. Ramsey, D. Orrell, and H. Bolouri, "Dizzy: stochastic simulation of large-scale genetic regulatory networks (supplementary material)," *J Bioinform Comput Biol*, vol. 3, pp. 437-54, 2005.

[48] L. You, A. Hoonlor, and J. Yin, "Modeling biological systems using Dynetica--a simulator of dynamic networks," *Bioinformatics*, vol. 19, pp. 435-6, 2003.

[49] M. Tomita, K. Hashimoto, K. Takahashi, T. S. Shimizu, Y. Matsuzaki, F. Miyoshi, K. Saito, S. Tanida, K. Yugi, J. C. Venter, and C. A. Hutchison, 3rd, "E-CELL: software environment for whole-cell simulation," *Bioinformatics*, vol. 15, pp. 72-84, 1999.

[50] S. Baigent, "Software review. Gepasi 3.0," *Brief Bioinform*, vol. 2, pp. 300-2, 2001.

[51] P. Mendes, "GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems," *Comput Appl Biosci*, vol. 9, pp. 563-71, 1993.

[52] P. Mendes, "Biochemistry by numbers: simulation of biochemical pathways with Gepasi 3," *Trends Biochem Sci*, vol. 22, pp. 361-3, 1997.

[53] P. D. Karp, S. Paley, and P. Romero, "The Pathway Tools software," *Bioinformatics*, vol. 18 Suppl 1, pp. S225-32, 2002.

[54] N. Le Novere and T. S. Shimizu, "STOCHSIM: modelling of stochastic biomolecular processes," *Bioinformatics*, vol. 17, pp. 575-6, 2001.

[55] A. M. Kierzek, "STOCKS: STOChastic Kinetic Simulations of biochemical systems with Gillespie algorithm," *Bioinformatics*, vol. 18, pp. 470-81, 2002.

[56] H. M. Sauro, M. Hucka, A. Finney, C. Wellock, H. Bolouri, J. Doyle, and H. Kitano, "Next generation simulation tools: the Systems Biology Workbench and BioSPICE integration," *Omics*, vol. 7, pp. 355-72, 2003.

[57] J. Schaff, C. C. Fink, B. Slepchenko, J. H. Carson, and L. M. Loew, "A general computational framework for modeling cellular structure and function," *Biophys J*, vol. 73, pp. 1135-46, 1997.

[58] J. C. Schaff, B. M. Slepchenko, and L. M. Loew, "Physiological modeling with virtual cell framework," *Methods Enzymol*, vol. 321, pp. 1-23, 2000.

[59] M. G. Smith, Jona, G., Ptacek, J., Devgan, G., Zhu, H., Zhu, X., and Snyder, M. , "Global analysis of protein function using protein microarrays," *Mechanisms of Ageing and Development*, vol. 126, pp. 171-175, 2005.

[60] C. L. Tucker, J. F. Gera, and P. Uetz, "Towards an understanding of complex protein networks," *Trends Cell Biol*, vol. 11, pp. 102-6, 2001.

[61] D. Thieffry, A. M. Huerta, E. Perez-Rueda, and J. Collado-Vides, "From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in Escherichia coli," *Bioessays*, vol. 20, pp. 433-40, 1998.

[62] J. R. Zavzavadjian, S. Couture, W. S. Park, J. Whalen, S. Lyon, G. Lee, E. Fung, Q. Mi, J. Liu, E. Wall, L. Santat, K. Dhandapani, C. Kivork, A. Driver, X. Zhu, M. S. Chang, B. Randhawa, E. Gehrig, H. Bryan, M. Verghese, A. Maer, B. Saunders, Y. Ning, S. Subramaniam, T. Meyer, M. I. Simon, N. O'Rourke, G. Chandy, and I. D. Fraser, "The alliance for cellular signaling plasmid collection: a flexible resource for protein localization studies and signaling pathway analysis," *Mol Cell Proteomics*, 2006.

[63] N. Le Novere, B. Bornstein, A. Broicher, M. Courtot, M. Donizelli, H. Dharuri, L. Li, H. Sauro, M. Schilstra, B. Shapiro, J. L. Snoep, and M. Hucka, "BioModels Database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems," *Nucleic Acids Res*, vol. 34, pp. D689-91, 2006.

[64] B. K. Hou, J. S. Kim, J. H. Jun, D. Y. Lee, Y. W. Kim, S. Chae, M. Roh, Y. H. In, and S. Y. Lee, "BioSilico: an integrated metabolic database system," *Bioinformatics*, vol. 20, pp. 3270-2, 2004.

[65] P. Pharkya, E. V. Nikolaev, and C. D. Maranas, "Review of the BRENDA Database," *Metab Eng*, vol. 5, pp. 71-3, 2003.

[66] I. M. Keseler, J. Collado-Vides, S. Gama-Castro, J. Ingraham, S. Paley, I. T. Paulsen, M. Peralta-Gil, and P. D. Karp, "EcoCyc: a comprehensive database resource for Escherichia coli," *Nucleic Acids Res*, vol. 33, pp. D334-7, 2005.

[67] A. Bairoch, "The ENZYME database in 2000," *Nucleic Acids Res*, vol. 28, pp. 304-5, 2000.

[68] M. Kanehisa and S. Goto, "KEGG: kyoto encyclopedia of genes and genomes," *Nucleic Acids Res*, vol. 28, pp. 27-30, 2000.

[69] R. Caspi, H. Foerster, C. A. Fulcher, R. Hopkinson, J. Ingraham, P. Kaipa, M. Krummenacker, S. Paley, J. Pick, S. Y. Rhee, C. Tissier, P. Zhang, and P. D. Karp, "MetaCyc: a multiorganism database of metabolic pathways and enzymes," *Nucleic Acids Res*, vol. 34, pp. D511-6, 2006.

[70] R. Overbeek, N. Larsen, G. D. Pusch, M. D'Souza, E. Selkov, Jr., N. Kyrpides, M. Fonstein, N. Maltsev, and E. Selkov, "WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction," *Nucleic Acids Res*, vol. 28, pp. 123-5, 2000.

[71] D. T. Gillespie, "Exact Stochastic Simulation of Coupled Chemical Reactions," *J Phys Chem*, vol. 81, pp. 2340-2361, 1977.

[72] N. Barkai and S. Leibler, "Robustness in simple biochemical networks," *Nature*, vol. 387, pp. 913-7, 1997.

[73] O. Kobiler, A. Rokney, N. Friedman, D. L. Court, J. Stavans, and A. B. Oppenheim, "Quantitative kinetic analysis of the bacteriophage lambda genetic network," *Proc Natl Acad Sci U S A*, vol. 102, pp. 4470-5, 2005.

[74] D. Bray, R. B. Bourret, and M. I. Simon, "Computer simulation of the phosphorylation cascade controlling bacterial chemotaxis," *Mol Biol Cell*, vol. 4, pp. 469-82, 1993.

[75] R. McDaniel and R. Weiss, "Advances in synthetic biology: on the path from prototypes to applications," *Curr Opin Biotechnol*, vol. 16, pp. 476-83, 2005.

[76] M. Sudol, H. I. Chen, C. Bougeret, A. Einbond, and P. Bork, "Characterization of a novel protein-binding module--the WW domain," *FEBS Lett*, vol. 369, pp. 67-71, 1995.

[77] A. Arkin, J. Ross, and H. H. McAdams, "Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected Escherichia coli cells," *Genetics*, vol. 149, pp. 1633-48, 1998.

[78] L. Mao and H. Resat, "Probabilistic representation of gene regulatory networks," *Bioinformatics*, vol. 20, pp. 2258-69, 2004.

[79] T. C. Meng, S. Somani, and P. Dhar, "Modeling and simulation of biological systems with stochasticity," *In Silico Biol*, vol. 4, pp. 293-309, 2004.

[80] M. B. Elowitz, A. J. Levine, E. D. Siggia, and P. S. Swain, "Stochastic gene expression in a single cell," *Science*, vol. 297, pp. 1183-6, 2002.

[81] P. S. Swain, M. B. Elowitz, and E. D. Siggia, "Intrinsic and extrinsic contributions to stochasticity in gene expression," *Proc Natl Acad Sci U S A*, vol. 99, pp. 12795-800, 2002.

[82] D. T. Gillespie, "Stochastic Simulation of Chemical Kinetics," *Annu Rev Phys Chem*, 2006.

[83] D. T. Gillespie, "A General Method for Numerically Simulating the Stochastic Time Evolution of Coupled Chemical Reactions," *J Comput Phys*, vol. 22, pp. 403-434, 1976.

[84] T. B. Kepler and T. C. Elston, "Stochasticity in transcriptional regulation: origins, consequences, and mathematical representations," *Biophys J*, vol. 81, pp. 3116-36, 2001.

[85] D. Gonze and A. Goldbeter, "Circadian rhythms and molecular noise," *Chaos*, vol. 16, pp. 026110, 2006.

[86] D. Gonze, J. Halloy, and A. Goldbeter, "Robustness of circadian rhythms with respect to molecular noise," *Proc Natl Acad Sci U S A*, vol. 99, pp. 673-8, 2002.

[87] J. Hubble, "Monte Carlo simulation of biospecific interactions," *Biotechnol Lett*, vol. 22, pp. 1483-1486, 2000.

[88] H. H. McAdams and A. Arkin, "Stochastic mechanisms in gene expression," *Proc Natl Acad Sci U S A*, vol. 94, pp. 814-9, 1997.

[89] C. a. A. Rao, A, "Stochastic chemical kinetics and the quasi-steady-state assumption: Application to the Gillespie algorithm," *J Chem Phys*, vol. 118, pp. 4999-5010, 2003.

[90] A. Chatterjee, K. Mayawala, J. S. Edwards, and D. G. Vlachos, "Time accelerated Monte Carlo simulations of biological networks using the binomial tau-leap method," *Bioinformatics*, vol. 21, pp. 2136-7, 2005.

[91] H. Salis and Y. Kaznessis, "Accurate hybrid stochastic simulation of a system of coupled chemical or biochemical reactions," *J Chem Phys*, vol. 122, pp. 54103, 2005.

[92] M. a. E. Bentele, R, "General Stochastic Hybrid Method for the Simulation of Chemical Reaction Processes in Cells," in *Computational Methods in Systems*

Biology, vol. 3082, *Lecture Notes in Computer Science*. Heidelberg: Springer Berlin/Heidelberg, 2005, pp. 248-251.

[93] E. L. Haseltine and J. B. Rawlings, "On the origins of approximations for stochastic chemical kinetics," *J Chem Phys*, vol. 123, pp. 164115, 2005.

[94] E. L. a. R. Haseltine, J.B., "Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics," *J Chem Phys*, vol. 117, pp. 6959-6969, 2002.

[95] J. Puchalka and A. M. Kierzek, "Bridging the gap between stochastic and deterministic regimes in the kinetic simulations of the biochemical reaction networks," *Biophys J*, vol. 86, pp. 1357-72, 2004.

[96] K. Burrage, T. Tian, and P. Burrage, "A multi-scaled approach for simulating chemical reaction systems," *Prog Biophys Mol Biol*, vol. 85, pp. 217-34, 2004.

[97] K. P. Hoeflich and M. Ikura, "Calmodulin in action: diversity in target recognition and activation mechanisms," *Cell*, vol. 108, pp. 739-42, 2002.

[98] A. Crivici and M. Ikura, "Molecular and structural basis of target recognition by calmodulin," *Annu Rev Biophys Biomol Struct*, vol. 24, pp. 85-116, 1995.

[99] G. Guntas, T. J. Mansell, J. R. Kim, and M. Ostermeier, "Directed evolution of protein switches and their application to the creation of ligand-binding proteins," *Proc Natl Acad Sci U S A*, vol. 102, pp. 11224-9, 2005.

[100] G. Guntas, S. F. Mitchell, and M. Ostermeier, "A molecular switch created by in vitro recombination of nonhomologous genes," *Chem Biol*, vol. 11, pp. 1483-7, 2004.

[101] R. W. Roberts, "Engineering switches, genetically," *Chem Biol*, vol. 11, pp. 1475-6, 2004.

Appendices

The appendices include the codes for the MBS program.

Appendix A

Data structure

Molecule information setup

`static int N_species;`	**number of species**
`static int* mol_total;`	**number of molecule per specie**
`static bool** mol_alive_new;`	**active flag of each molecule in the new time step**
`static bool** mol_alive;`	**active flag of each molecule in the current time step**
`static int* mol_totalAlive_new;`	**number of active molecule per specie in the new time step**
`static int* mol_totalAlive;`	**number of active molecule per specie in the current time step**
`static double*** mol_pos;`	**position of molecule**
`static double* mol_mass;`	**mass of molecule**
`static double* mol_size;`	**diameter of molecule**
`static double* mol_diffConst;`	**diffusion coefficient**

`static char** mol_name;`	**name of molecule**
`static int* mol_region;`	**valid region the molecule can diffuse in**

Environment setup

`static double temperature;`	**initial temperature**
`static double viscosity;`	**viscosity of solution**
`static double pH;`	**initial pH**

Reaction setup

`static int N_reactions;`	**number of reactions**
`static int N_fwd;`	**number of association reactions**
`static int N_rev;`	**number of dissociation reactions**
`static int N_rxnID_fwd;`	**number of association reaction constructs**
`static int N_rxnID_rev;`	**number of dissociation reaction constructs**
`static int fwd_max_rxn;`	**max number of association reaction per reagent**

```
static int rev_max_mol;            max number of dissociation
                                   reaction per reagent

struct ForwardReaction
{
     int reagent;
     int partner;
     int product;
     double kf;
};
ForwardReaction* p_fwd;
int*** ppp_fwd_hash;

struct ReverseReaction
{
     int reagent;
     double kr;
     int num_prod;
     int* product;
};
ReverseReaction* p_rev;
int** pp_rev_hash;

static double** V0_v;              effective volume
```

Simulation setup

```
static int total_step;             total step of simulation
static int current_step;           current step index
```

```
static double time_step;            step length in time
```

Check data structure initiation

```
static bool data_init_success=false;    whether data structure
                                        initiation is
                                        successful
```

Appendix B

Function definitions

dataFetcher.h

Interface between the data structure and the function calls in the program.

```
//allocates memory
void initDataStructure_fromFile(char* filename);

//clears memory must pair with initDataStructure_fromFile
void deleteDataStructure();

//using SQL database, not implemented
void initDataStructure_fromSQL();
bool data_initialization_success();

//storing data
void snapshotSetup_toFile(char* filename);
void snapshotMol_toFile(char* filename);
void snapshotRxn_toFile(char* filename);
void output_to_SQL();

//movie reconstruction
void createMovie_toFile(char* movie_filename, int stepNumber);
void readMovie_fromFile(char* movie_filename, fpos_t*
                        filePosition);

//data output
void createData_toFile(char* data_filename, int stepNumber);

//new time step handling
void copy_current_new();

//replacing the new time step with the current
```

```
void swap_current_new();

//check if a particular molecule is active in new step
bool alive_new(int speciesID, int index);

//set status of the molecule in new step
void setAlive_new(int speciesID, int index, bool fate);

//active population in new step
int alivePopulation_new(int speciesID);

//find the index of the first inactive molecule
int findFirstDeadIndex_new(int specieID);

//molecule information
int numSpecies();
int speciePopulation(int speciesID);
bool alive(int speciesID, int index);
void setAlive(int speciesID, int index, bool fate);
int alivePopulation(int speciesID);
int totalAlivePopulation();
void position(int speciesID, int index, double* x, double* y,
              double* z);
void setPosition(int speciesID, int index, double x, double y,
                 double z);
double mass(int speciesID);
double size(int speciesID);
double diffConst(int speciesID);
void name(int speciesID, char* output);
int specieRegion(int specieID);
int findFirstDeadIndex(int specieID);
int findFirstAliveIndex(int specieID);
```

```
//environment information
double readTemperature();
void setTemperature(double T);
double readViscosity();
void setViscosity(double v);
double readPH();
void setPH(double p);

//reaction info, FR-forward reaction, RR-reverse reaction
int totalRxn();
int totalRxn_F();
int totalRxn_R();
int numRxn_F(int specie1, int specie2);
int numRxn_R(int specie);
int totalrxnID_F();
int totalrxnID_R();
rxnID rxnID_F(int specie1, int specie2, int pick);
rxnID rxnID_R(int specie, int pick);
double kf(rxnID id);
double kr(rxnID id);
int numProd_R(rxnID id);
int partnerID_F(rxnID id);
int productID_F(rxnID id);
void productID_R(rxnID id, int* products, int N);
double V0(int specie1, int specie2);

//simulation information
int totalStep();
int readCurrentStep();
void setCurrentStep(int scs);
double dt();
```

```
//error handling
void errorOut(char* message);

//distance
double dist2Mol(int specieID1, int index1, int specieID2, int
                index2);
double dist2Mol2(int specieID1, int index1, int specieID2, int
                 index2);
double dist3D(double x1,double y1,double z1,double x2,double
              y2,double z2);
double dist3D2(double x1,double y1,double z1,double x2,double
              y2,double z2);
bool withinDist(int specieID1, int index1, int specieID2, int
                index2, double dist);
```

diffusion.h

Handles molecule diffusion in the correct regions

```
//this function takes pointer to values x,y,z and change their
values based on D and t using dPDF
void diffuse(double* x,double* y,double* z,double time,double
             D,int region);

//diffuse all molecules
void diffuseAll();

//these functions set the upper bound of the range of
molecules that can react (2x half max half width)
double max_diffusion_radius2(double D, double time);
```

```
double max_diffusion_radius(double D, double time);
```

experiment.h

Handles adding and removing molecules in different time steps

```
//initializing experiment information
void initExperiment_fromFile(char* filename);
bool exp_initialization_success();

//reprinting experiment information
void snapshotExperiment_toFile(char* filename);

//perform experiment
void experiment(int ts);
```

geometry.h

Initialize molecule positions and check for boundary conditions

```
//parse boundary/region information from file to region data structure.
void initGeometry_fromFile(char* filename);
bool geo_initialization_success();

//reprinting experiment information
void snapshotGeometry_toFile(char* filename);

//check if point is in geometric shapes
bool inBox(double cx, double cy, double cz, double wx, double
          wy, double wz,  double x, double y, double z);
```

```
bool inBoxShell(double cx, double cy, double cz, double wx1,
                double wy1, double wz1,  double wx2, double
                wy2, double wz2, double x, double y, double z);
bool inSphere(double cx, double cy, double cz, double radius,
              double x, double y, double z);
bool inSphereShell(double cx, double cy, double cz, double r1,
                   double r2, double x, double y, double z);
bool inCylinder(double cx, double cy, double cz, double r,
                double l, double x, double y, double z);
bool inCylinderShell(double cx, double cy, double cz, double
                     r1, double r2, double l, double x, double
                     y, double z);
```

//returns what region is (x,y,z) in

```
bool inRegion(double x, double y, double z, int regionID);
```

//create a point (x,y,z) in a specified region

```
void createToRegion(int regionID, double* x, double* y,
                    double* z);
```

inverse3DGaussian.h

Numerical function for inverse 3D Gaussian function

//loads the inverse 3DGaussian function data from file

```
bool init_inverse_3DGaussian();
```

//this function takes input from 0 to 1

```
double inverse_3DGaussian(double x);
```

randomNumber.h

Handles all random number generation in the simulation

```
//basic random number generator
void initRand();
double random_number(double lower, double upper);
int random_integer(int lower, int upper);

//3D isotropic density radius distribution function
double random_radius_3Dgaussian(double D, double time);
double random_radius_3Dhomogeneous(double lower, double upper);

//2D isotropic density radius distribution
double random_radius_2Dhomogeneous(double lower, double upper);

//3D isotropic spherical angles
double random_angle_phi();
double random_angle_theta();

//randomize molecule order
void randomize_MO();
int length_MO();

//read Molecule Order
void read_MO(int n, int* specie, int* ID);
//clear the memory allocated for MO
void delete_MO();

//randomize reaction order
void randomize_RO(int specie);
int length_RO();
```

```
//read reaction order
void read_RO(int ro_index, bool* isFwdRxn, rxnID* id);
void delete_RO();

//random position generators
void randBox_BBB(double x1, double y1, double z1, double x2,
                 double y2, double z2, double* x, double* y,
                 double* z);
void randBox_CW(double cx, double cy, double cz, double wx,
                double wy, double wz,  double* x, double* y,
                double* z);
void randBoxShell_CWW(double cx, double cy, double cz, double
                      wx1, double wy1, double wz1,  double
                      wx2, double wy2, double wz2, double* x,
                      double* y, double* z);
void randSphere(double cx, double cy, double cz, double radius,
                double* x, double* y, double* z);
void randSphereShell(double cx, double cy, double cz, double
                     r1, double r2, double* x, double* y,
                     double* z);
void randCylinder(double cx, double cy, double cz, double r,
                  double l, double* x, double* y, double* z);

void randCylinderShell(double cx, double cy, double cz, double
                       r1, double r2, double l, double* x,
                       double* y, double* z);
```

reaction.h

Calculates the reaction probabilities

```
//reaction probability between two molecules
double rxnProbablity(int specieID1, int specieID2,  rxnID r,
                     double l);

//association reaction probability
bool reactFWD(int specie1, int index1, int specie2, int index2,
              rxnID r);

//dissociation reaction probability
bool reactREV(int specie, int index, rxnID r);

//decides the reaction for all molecules in the volume
void reactAll();
```

Appendix C

Main function structure

The main function handles the flow of the program and the data I/O.

```
//loading the libraries
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

#include "dataFetcher.h"
#include "diffusion.h"
#include "experiment.h"
#include "geometry.h"
#include "inverse3DGaussian.h"
#include "reaction.h"
#include "randomNumber.h"

//local function definition
void displayTimeLapse();

/*expected input format:
     MCR.exe setup experiment geometry -m movie -d data
     MCR.exe setup experiment geometry -m movie
     MCR.exe setup experiment geometry -d data
*/

//main program starts
int main(int argc, char **argv)
{
     //checking input parameters and ready for file inputs
     printf("\n\nTrying to understand what you want to do...");
     ...
```

```
...
errorOut("movie filename exist! Change movie filename!");

//data base initialization
initRand();
...
...
//snapshotRxn_toFile("a");

//simulation starts here, going through all steps sequentially
for(int i=0;i<totalStep();i++)
{
     //set the current time step index
     setCurrentStep(i);

     //changing reaction conditions: init/add molecules, change temperature etc...
     experiment(readCurrentStep());

     //diffuse the molecules
     diffuseAll();

     //this react puts all info into the "new" mol_alive_new, mol_totalAlive_new, the old mol_alive, mol_totalAlive are unchanged.
     reactAll();

     //create movie
     if(outputMovie)
          createMovie_toFile(nameMovie,i);
```

```
            //create datafile
            if(outputData)
                createData_toFile(nameData,i);

        }

        //clear memory data structure
        deleteDataStructure();

        //display the time took for simulation to complete
        displayTimeLapse();

        return 0;
}
```