NAGOYA UNIVERSITY

Swarm Intelligence-Based DNA Computation

Rizki Mardian

Department of Micro-Nano Systems Engineering

Nagoya University

A thesis submitted for the degree of

Doctor of Engineering (Dr.Eng.)

2016

To a little boy who imagined so many things in the past,
both of his parents who he always made worried with everything he had done,
all teachers who had entrusted so much knowledge to him,
and the dreams which are yet still unnamed...

Acknowledgements

When I embarked my journey towards the long-romantic odyssey called the graduate school, I did not know what to expect. Half of me decided to pursue a higher degree of education because I was not ready to enter industry by that time, while the rest always dreamed about experiencing life in a new culture, in a place that was far away from home. Then, I decided to come to Japan to study about "robotics". I thought I would.

What actually happened during the last five years were a bit of turning points in my life. First, I was trained as a pure computer scientist and I thought that I would always be. I had never expected to switch my field to something-biological, as it was one of the subjects I hated the most in high school. I never thought I would fall in love with it, but I am happy to be proven wrong. Second, I had never seen myself as a scientist. For the past-me, doing a research was merely an inevitable process in obtaining an academic degree. Yet, through the up and down of the past few years, I started to see something else. I started to see research as "art" in science and technology; it is not about how we can make as hardest as possible things in this world, but instead to get as simplest as possible new ideas worked. And I think I found my calling there.

It is with my highest humbleness I would like to express my gratitude to my three advisers, who made all of those things possible: Prof. Yasuhisa Hasegawa, Dr. Kosuke Sekiyama, and Prof. Toshio Fukuda. Prof. Fukuda was the first reason I can find my way to graduate studies. The letter of acceptance of his had marked my scientific journey at Nagoya University. I was brought to the field of molecular computing and DNA nanotechnology by Dr. Sekiyama. Having been working directly under his supervision, his supports throughout my doctoral courses are not shortchanged. His suggestions, criticisms, and advices were essential to keep things in perspective. Prof. Hasegawa was there since I was a D2. His new way of leadership had brought up some motivations in the turbulent years, and let me pick things that I have might lost along the way towards the accomplishment of my doctoral degree.

I would like to extend my gratefulness and appreciation to members of Kiban-S Molecular Robotics Project of Japan, for all meaningful discussions, both remote and occasional face-to-face meetings. It is always not easy when you work with something new, have no expertise in the field, and be in a relatively small team. I have acquired some valuable advices and suggestions that help me survive and progress in my work. Especially, I would like to thank Prof. Satoshi Murata, for having provided me the opportunity of trainings in his lab. I was indebted to Dr. Ibuki Kawamata, Keitel Cervantes, Tomoyuki Hirano, Ryo Kageyama, and members of Murata-Kawamata/Nomura's lab who had taught me the experimental procedures.

I thank all my lab-mates, from the three different periods of Fukuda's lab to Sekiyama's lab to Hasegawa-Sekiyama's lab. Adi Wibowo, and Noritaka Mori, the fellows of our small DNA computing group. It has been an enlightening experience working alongside each other, in a new field that all of us had no idea in the beginning. Dr. Masahiro Nakajima, Dr. Masaru Kojima, and Dr. Hirotaka Tajima, all the assistant professors of the lab. They have provided me with the early training of wet-lab protocols. Zeyang Liu, my table-neighbor and a fellow that shares common vision with me, both in science and gym. Thanks for those funny and desperate talks and all dinners you have treated me. Do not get so stressed! Former members who graduated earlier: Leandro Hideo Iha, and Takayuki Umeda, both were the tutors who helped me get adjusted with life in Japan during my first years. Dr. Tao Yue, Dr. Chengzhi Hu, Dr. Yajing Shen, Dr. Fei Chen, thank you for the valuable discussions during your time in lab. And all other fellows that I cannot list one by one here.

My special thanks I should address to Dr. Paul Rothemund and Dr. Rizal Hariadi. I cannot describe how lucky I was to be able to meet Dr. Rothemund during the summer-camp of the molecular robotics group in 2012. It had significantly changed my life and had shaped my new dream as a scientist. He later introduced me to Dr. Hariadi, who was in the following years had countlessly helped, motivated, and supported me in every way that he could. Thank you for letting me taking your valuable time in proofreading my letters, having skype-meeting, and even provided me with shelter when I was around. I look up to both of you, and I wish that one day I can be as bright as you guys in science and technology.

I also thank my thesis committee members: Prof. Fumihito Arai, and Dr. Yasuhiro Suzuki, for review and comments.

Last but not least, I am very grateful to the MEXT: Ministry of Education, Culture, Sports, Science, and Technology in Japan, not only for providing me with stipends, but also for the chance being a proud member of fellow communities who are as excited by academics and science as I am. It is also thanks to the invaluable encouragement of my undergraduate adviser, Dr. Wisnu Jatmiko who had pushed and encouraged me in applying for this scholarship. I would love to mention all the MEXT 2010 recipients as well, those who once were strangers then became family: Rizki Andini, Rina Khairani, Agro Rachmatullah, Marissa Munif Hassan, Wulan Rahmadhany, Heni Hernawati, and Ermida Simajuntak. Lets burn some calories again running all the way in another airport some time!

Finally, I presented and dedicated this work to both of my parents, Rahima Adnan and Mahyuddin Taher. They deserve very special thanks for their unconditional love and putting up with me (and I know they will always be). Both my brother and my sister, Rahmad Mahendra and Rahma Muthia, I know I am not good at keeping in touch at a distance, and definitely I am very much terrible in exemplifying my feelings too (you know, I had a long pause writing this part), but there is no word I can use to describe how fortunate I am being born as the second of the RM-brotherhood.

Abstract

This thesis is a summary of four independent but related manuscripts on computational design of synthetic bio-molecular systems based on DNA strand displacement reaction.

In the first part, we report the design of the DNA-based circuit in well-mixed chemical systems. Chemical Reaction Networks is employed as the molecular programming principle, and Immune Network Theory as the problem solving algorithm.

In the second part, Ant Double-Bridge System is chosen as a cue in designing the spatially localized DNA architecture based on computation with molecular walkers.

In the third part, we discuss the model-based coordination strategy for DNA-based agents based on Petri Nets. Preliminary experimental results of the DNA-based Petri Nets operators are presented as a proof-of-concept of the designed model.

In the last part, the in-vitro implementation of the DNA-based Finite State Machine is reported. Furthermore, a design of probabilistic DNA gate is proposed, to simulate stochastic-like computation based on bio-molecular reactions.

Contents

List of Figures

1

Introduction

1.1 Overview

Over the past few decades, DNA has been used as a non-genetic or non-biological engineering material to develop various interesting applications at the molecular level (3, 4, 5, 6, 7, 8). A toehold-mediated strand-displacement and branch-migration mechanism has been treated as a promising methodology to deliver DNA-based dynamical systems, which enables the potential of the bio-chemical circuits to approach complexities of silicon-based ones (1) (Figure 1.1).

Numerous independent functions have been proposed on this mechanism, including: chemical reaction networks (9, 10, 11), logic circuits (12, 13, 14, 15, 16, 17), nano-motors and synthetic walkers (18, 19, 20, 21, 22, 23, 24). Altogether, these molecular implementation can exhibit intricate systems with growing complexities, such as molecular robotics (25, 26). For example, the DNA walker is designed to follow predetermined paths like locomotion of the autonomous robot (20, 22). The unique thing is that the behavior of the mobile DNA is not explicitly programmed, but it is driven by the peculiar structure interacting with the environment. This indicates the capability of the information sensing as well. In other reports, the capability of a tweezer-shaped and a forklift-shaped DNA to interact with other biological materials opens the possibility for autonomous actuation in the molecular level (23, 24). In the future, this direction is expected to support the discovery of new therapeutics methods, such as intelligent drugs-delivery.

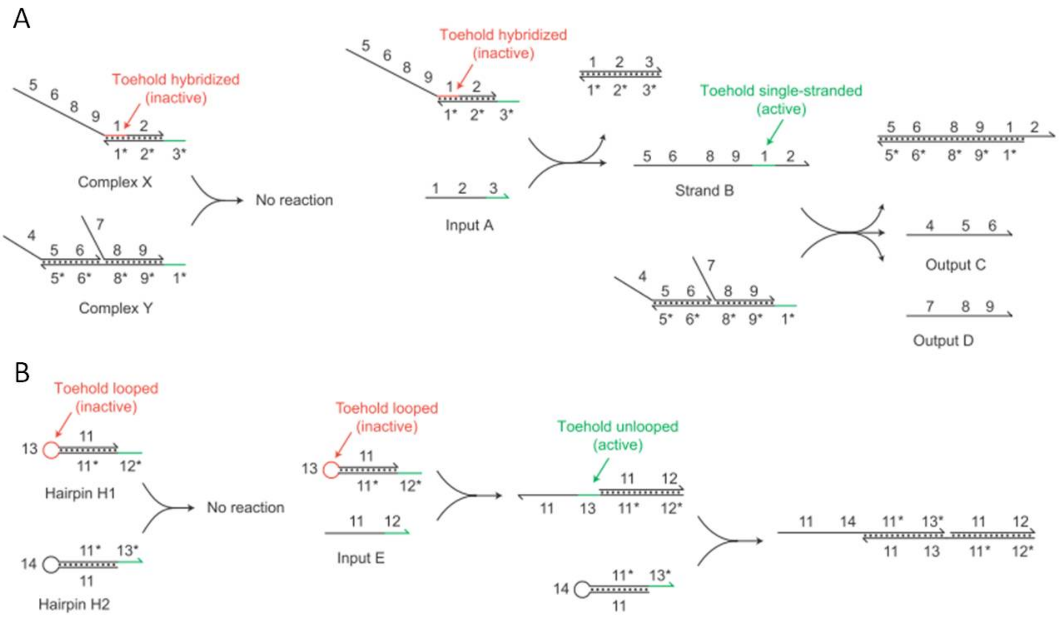


Figure 1.1: Basic of DNA strand displacement reactions. A) Toehold exchange mechanism. B) Hairpin-based mechanism. Image was taken from (1).

Despite of its importance, there are still not so many literature and references that address and explore the issue of the computational design of DNA-based systems. In this work, we focus on designing and modeling of DNA-based information processing technology based on DNA strand displacement reaction that is based on nature-inspired computation, the principle of distributed and interaction-based model, in order to develop functions as simple as logical operation (e.g. digital logic circuit) and as complex as stochastic decision-making and learning scheme. This is expected to introduce a new ability to bio-molecular systems to compute and to offer a certain degree of autonomy to the DNA-based molecular system.

One obstacle in developing DNA-based information processing systems is how to perform mathematical operation and logical control over DNA reactions. While numerous designs of DNA-based logic gates have been presented, performing complex computation still remains a big challenge, due to the limitation of the synthetic bio-molecular systems in terms of computation speed and error-rate. Nature-inspired computation and self-organizing model can be utilized to over-come this problem. In this work, the implementation of a DNA strand displacement system based on nature-inspired computation is observed. We specifically worked on Immune Network Theory, which is inspired by the principle of distributed interaction of humans natural immune system. By using mass action kinetics model, the compilation of its DNA-based operations is derived from the existed mathematical model. This DNA-based implementation is then compared to the silicon-based programming. From the obtained results, we can see positive correlations that our DNA-based Immune Network model satisfies the behavior of the mathematical model. In robotics and applied math fields, Immune Network Theory has been utilized to solve various computational problems, including decision-making and reinforcement learning. Thus, we expect that we can also introduce the similar capabilities to DNA-based machines by using our model.

A major challenge in the development of evolvable, autonomous, and programmable bio-molecular machines is the introduction of the ability to cope with external stimuli. The intriguing question is: is it possible to build artificial molecular systems that can learn to adapt their environmental condition? In this work, DNA strand displacement was chosen as the main framework for modeling a DNA circuit capable of complex computational mechanisms such as decision-making and reinforcement learning. The design is inspired by positive and negative feedback mechanisms in simple systems, such as the collective food foraging of ants colony. We show how it may be used as the basis for designing pathways of DNA reaction systems; which exploits swarm interaction between ant-inspired agents, spatial and temporal aspects of the designed circuit, as well as indirect communication mediated by the environment. Our model is compatible to recent architectures, termed spatially localized DNA circuits. In contrast with free-floating systems, DNA complexes are immobilized on nanostructures, which in principle speeds up the reaction and increases the efficiency. The correctness of our model was verified in silico via quantitative measurement of reaction kinetics. Our results indicate that the circuit adaptively responds to the present stimuli, regardless of the initial conditions (in contrast to the currently available DNA strand displacement systems). The potential applications of our model include decision-making capable machine, and reusable DNA circuits.

1. INTRODUCTION

Coordination is an important aspect in developing distributed autonomous systems. In silicon-based agents, designing individual-level behavior that may emerge into one global function is a typical approach to such systems. Meanwhile, in DNA-based agents, programming of each individuals behavior still remains a big challenge, as they occur immediately after all reactants have been mixed into the solution. We strive to solve this challenge by proposing a novel architecture of interacting DNA-based molecular agents. This may be met through discretization of bio-molecular reactions into event-driven systems by using Petri Nets model (in contrast to natural chemical systems that are continuous and time-evolved). First, computational primitives based on DNA strand displacement reaction are introduced. Second, their molecular implementation is abstracted by Petri Nets for high-level design. Third, we propose the model of interacting multi-agent systems based on DNA-only reactions. We verify our design via computer-based simulation and show the initial experiments of Petri Nets operators (due to the limitation of currently available technologies, we were not able to fully tested the DNA-based Petri Nets model as it requires single molecule preparation). From the obtained results, we argue that our design strategy is feasible for coordinating interaction of distributed DNA-based systems.

Molecular robotics and other autonomous molecular machines, like their mechanical counterparts, are expected to perform intelligent tasks under minimum external supervisions. One strategy to accomplish such complex design is by representing internal states of the machines by using Finite State Machine. The transition between states is triggered by external stimuli. This can be an input signal from external systems, data acquired from the environment, or communication signals with other machines in the case of multi-agent systems. While there have been many proposals on how to implement deterministic transitions by DNA reactions, e.g. by DNA strand displacement cascades, the experimental procedure still remains a challenge. Moreover, in this work, we also propose a new design for stochastic transitions, allowing selection of transition in case of two or more possible next states. Our gate design is inspired by the principle of competing-for-limited-resources and the cooperative hybridization. Further application of such logic gate may also be applied to arbitrary stochastic DNA computation.

In general, our contributions can be divided into two: first is to develop a new DNA-based algorithm that is based on swarm-intelligence computation, and second is to bridge the current DNA computational system to any existed mathematicals and information technology models so that in the future a gap between molecular and traditional computing can be minimized.

1.2 Background

1.2.1 DNA Nanotechnology

Nanotechnology is a science and engineering to manipulate matters at the molecular scale, sized from 1 to 100 nanometer (1 nanometer $= 1 \times 10^{-9}$ meter). It ranges from many disciplines of researches, including material science, organic chemistry, molecular biology, semiconductor physics, and so on (27). By DNA nanotechnology, it means the nucleic-acid structure is manipulated to design and manufacture artificial systems that are intended for other technology purposes; rather than as a medium to carry the genetic information. It takes advantages from the strict base pairing rules of nucleic acids, which cause the portions of strands with complementary base sequences to bind together to form strong, rigid double helix structures. This allows the rational design of base sequences that will selectively assemble to form complex target structures with precisely controlled nanoscale features. Even though other nucleic acids, such as RNA and PNA can also be engineered, within the past few years, DNA has been the dominant material, leading to the use of the name DNA nanotechnology to describe the field.

The first scientist who is known as the father of DNA nanotechnology is Nadrian C. Seeman from New York University, for his attempt to construct three dimensional lattices made from DNA in the early 1980s (3). In 1991, he successfully realized the first synthesis of the three-dimensional nanoscale object: a cube made of DNA, from which he was awarded with Feynman Prize in nanotechnology years later (28). To this date, many designs have been developed; both static structures such as two and three dimensional crystal lattices, nanotubes, polyhedra, and arbitrary shapes (4, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41);

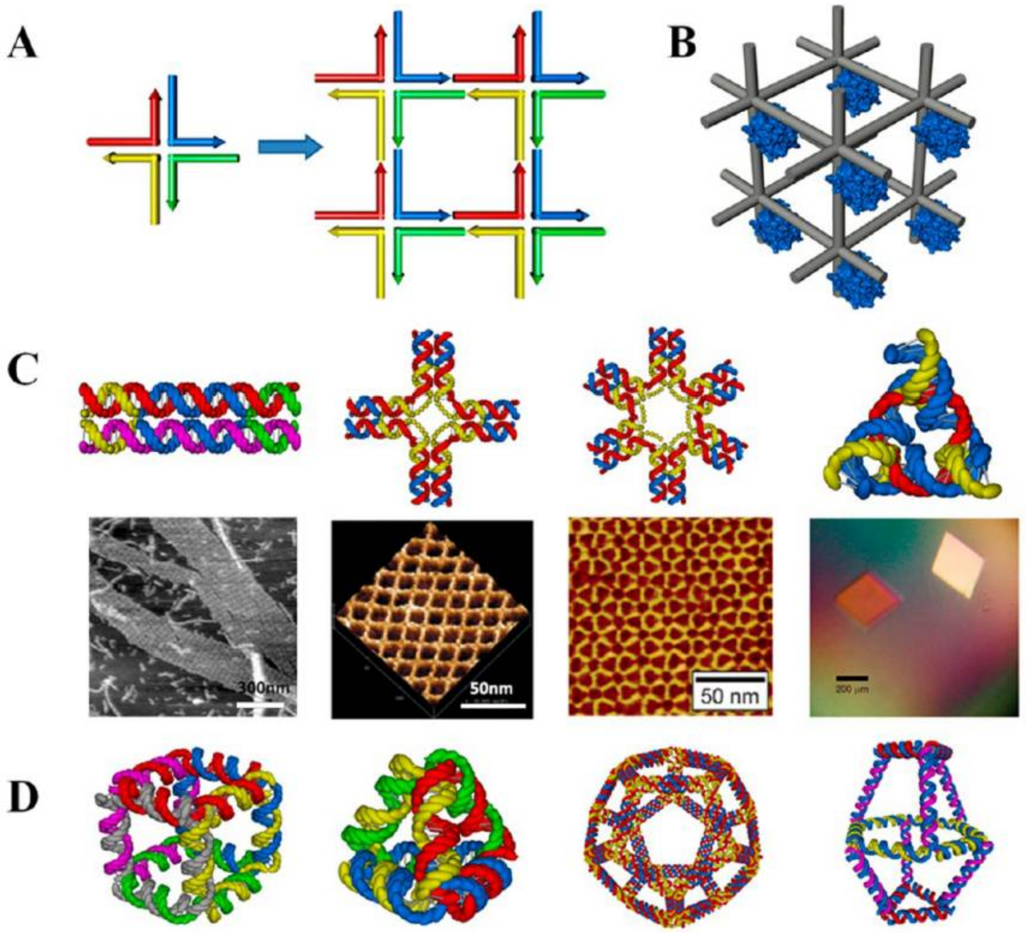


Figure 1.2: Recent advancements from the structural DNA nanotechnology. Image was taken from (2).

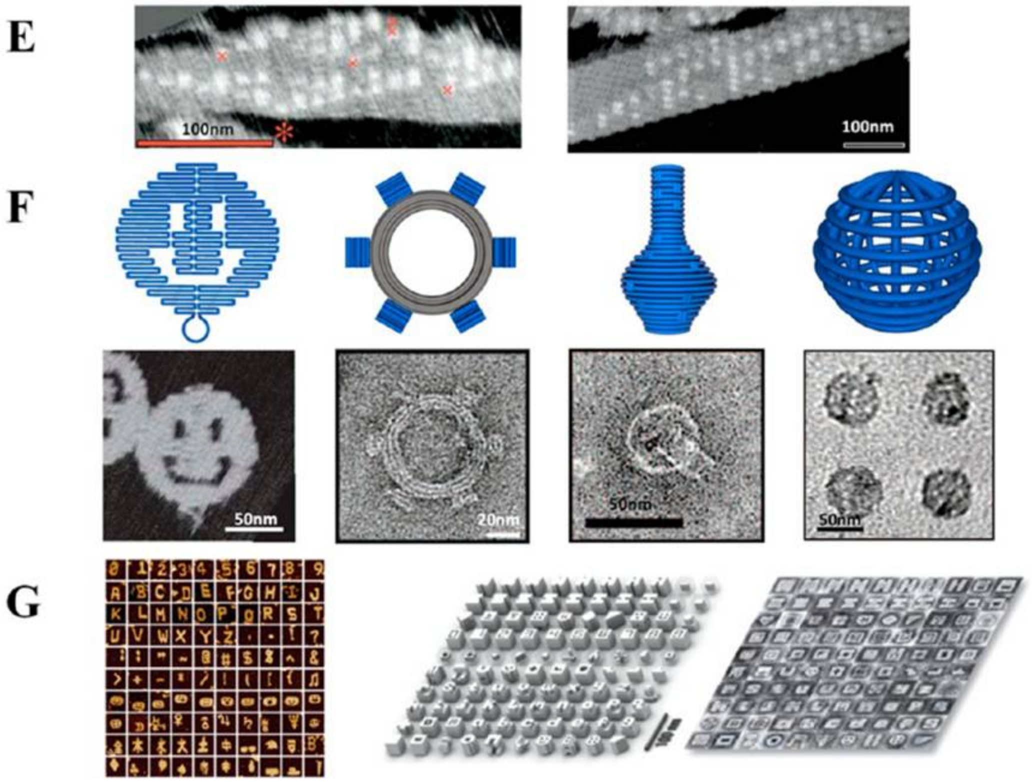


Figure 1.3: Recent advancements from the structural DNA nanotechnology. Image was taken from (2).

and functional devices such as reconfigurable structures, molecular walkers, and DNA computers (12, 13, 22, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53).

Numerous assembly methods are investigated, including tile-based structures to assemble from smaller structures, long single strand scaffold and shorter staple strands together by using the DNA origami, and dynamically reconfigurable structures such as toehold mediated branch migration and strand displacement. This field has then become the base for further advancement to other domains, such as DNA-based computation, DNA nanorobotics, and so on. Figure 1.2 and 1.3 show various DNA-based nanostructures that have been developed in the field of structural DNA nanotechnology until recent years.

1.2.2 DNA-Based Computation

DNA computing, or sometimes also referred to as molecular programming, is a form of computing in which DNA structure is used to carry data and information, during the process of computation. The basic principle of programming by DNA is to exploit the programmability of DNA strands based on the specific Watson-Crick binding of DNA bases, Adenine (A), Cytosine (C), Guanine (G) and thymine (T). Typically, A binds with T, while C binds with G. Hence, the sequence CGTACG hybridizes to its reverse complement (which often reversely written) CGTACG to form a double strand DNA. These sequences can then be designed in systematic ways to achieve particular behavior. For example, suppose there are three molecules of A, B, and C. We can design such that the first part of A is complementary to the last part of C, the last part of A is complementary to the first part of B and the last part of B is complementary to first part of C. In a maintained chemical condition, these molecules can assemble into a 3-way DNA junction. Even another design of 4-way DNA junction can be achieved by adding a molecule D, by following the same rules. Figure 1.4 summarizes about this principle.

The beginning of the DNA computing was marked by the work of Leonard Adleman in 1994 (54). By demonstrating a proof-of-concept by using DNA as a form of computation, he solved the seven-point Hamiltonian path problem through wet-lab experiments. This work was then extended by Richard Lipton

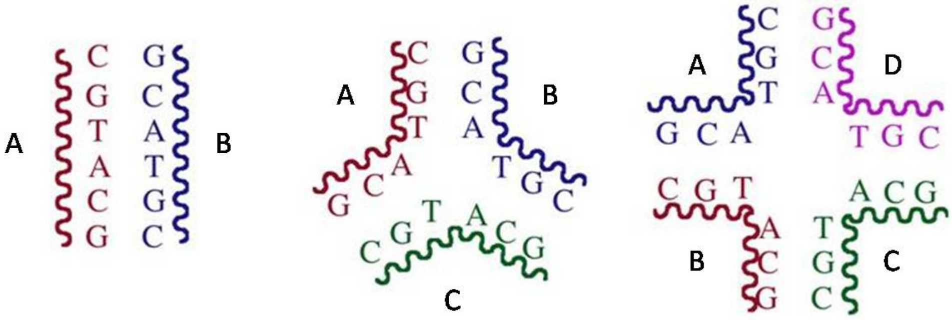


Figure 1.4: Schematic representation of how DNA molecules can form a double-stranded DNA, a 3-way junction or a 4-way junction

in 1995 (55), generalizing that this method can be useful to tackle NP-hard problems, which can hardly be solved by using a normal computer. Since then, DNA has been argued as the promising material to develop an alternative computer or other computing-based device in the future.

Despite of this breakthrough, in recent years, the Adleman's method found difficulties to progress; as this technique is prone to error or requires many experimental procedure which makes DNA-based computers practically in vain. Therefore, lately, the direction of the DNA-based computation research is rather to find a new way to control bio-molecular mechanisms for other engineering purposes in the interface between computer science and other nanotechnologies. There are various different methods in molecular programming, such as:

1. Enzymes. Enzyme based DNA computers are usually of the form of a simple Turing machine. It is actually analogous to the hardware, in the form of an enzyme, and the software, in the form of DNA. Example from this work: Benenson and colleagues have demonstrated a DNA computer using the FokI enzyme (56) and expanded on their work by going on to show automata that diagnose and react to prostate cancer (57). The disadvantage of this approach is since the design of the enzyme itself is also important, it is not suitable to build a large scale application.

2. Toehold exchange. DNA computers have also been constructed by using the concept of toehold exchange. An input DNA strand binds to a sticky end

> (or toehold) on another DNA molecule, which allows it to displace another strand segment from the molecule due the difference and competitiveness of entropy. There are a lot of work related to this approach, for example the creation of modular logic components such as AND, OR, and NOT gates and signal amplifiers, which can be linked into arbitrarily large computers. This class of DNA computers does not require enzymes or any other chemical capability or design from the DNA itself (12).

In this thesis, we research about the toehold exchange method towards DNA-based dynamical systems. Our objective is to deliver a complex system based on DNA reactions capable of "computation". The detail of the implementation, as well as the explanation about DNA strand displacement and toehold exchange based computation will be explained in later sections. While in structural DNA nanotechnology, attentions have been paid in designing various intricate two and three dimensional structures, researches in dynamic DNA nanotechnology focus on the non-equilibrium behaviors of the DNA-based systems which go through structural changes in the addition of external fuels. Figure 1.5 shows major domains in this area.

1.2.3 DNA Nanorobotics

The aim of DNA nanorobotics is to design and to fabricate of dynamic DNA nanostructures that perform specific tasks via a series of states changes. At the most references available so far, this task involve some form of robotics motion, such as locomotion or conformational changes. These states changes can be autonomous, in which case systems switch states without any external intervention; or non-autonomous, where some amount of specific species, such as DNA strands or enzymes, are introduced to enforce the process.

Various challenges arise in attempting to create a DNA nanorobot. The design of the DNA robot begins at the domain level where the overall mechanisms of the robot's actions are planned without actually assigning DNA sequences to the strands. Instead the different interacting segments of the DNA strands that "act" as the robot are assigned domain names which in the next step are assigned to

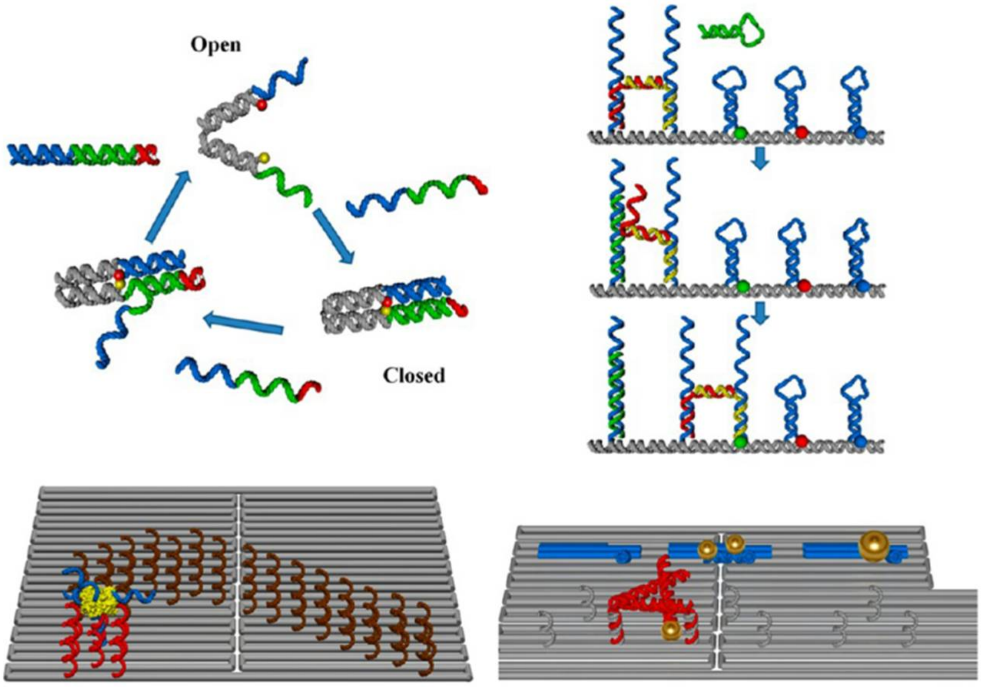


Figure 1.5: Recent advancements from the dynamic DNA nanotechnology. Image was taken from (2).

specific DNA sequences. The mapping of domains to DNA sequences can be done by taking care of spurious interaction among the various domains.

Another important consideration at this stage is the fuel that powers the robot. Typically, robots are powered either by enzymes that act upon specific DNA strands of the robot's components, or by the energy of hybridization of freely floating single stranded fuel DNA. Sometimes entropic effects can be used to power the states changes. For example, two DNA strands that are held together by the hybridization of a small domain might denature spontaneously leading to an increase in the entropy of the system.

Other challenges include actual assembly of the DNA nanorobot and its purification, setting up initial operating conditions and finally designing experiments that validate the proposed mechanism of action of the DNA nanorobot. It is very hard to directly observe the operation of the robot, therefore other mean of real time detection methods are often used, such as FRET, and so on.

Below are the list of reasons why DNA is a material uniquely suited for building and manipulation at the molecular scale. From the perspective of design, the advantages are: (1) The predictable behavior can be achieved by carefully programming the interaction of DNA sequences. (2) The basic geometric and thermodynamic properties of the double strand DNA are well understood and can be predicted by available software systems from relevant parameters like sequence composition, temperature, and solution composition. (3) Design of DNA nanostructures can be assisted in silico. To design DNA nanostructures or devices, one needs to design a library of single strand DNA strands with specific segments that hybridize to (and only to) specific complementary segments on other single strand DNA.

In the end, the ultimate goal of the DNA nanorobotics research is to create an autonomous machine working at the molecular level. This is interesting, since working at nanoscopic domain is still complicated so far, and it may open many possibilities of technology advancements in the future. By autonomous machines it has an analogous meaning to the traditional robots, that the molecular robots should possess the capability to interact with their environment, and to utilize those information to make an appropriate action without any intervention from outside resources. Since developing this kind of machine at the nanoscopic level by using silicon-based materials will be very difficult, the usage of the bio-materials such as DNA became a promising proposal. The idea is, instead of manufacturing the robots from the scratch, the self-assembly property of the biochemical systems can be utilized. The challenge is how to direct the interaction of those biomolecular components to achieve certain purpose as we intend.

1.3 Scope and Objectives

The DNA nanotechnology is an emerging study with a broad scope that crosses many different disciplines, such as: bio-chemistry, molecular bio-physics, mathematics, computer science, and so on. As mentioned earlier, generally, directions of this field can be divided into two: structural DNA nanotechnology, and dynamic DNA nanotechnology. The work presented in this thesis is relatively close to the second mentioned. In this research, we particularly focus on the computational design and information processing strategy based on a particular bio-molecular mechanism termed DNA strand displacement. The substantial challenge is how to enable computation (as simple as logical control, and as complex as decision-making and learning capability) through the DNA-based reaction. Further applications of the designed systems include molecular robotics and DNA-based computer.

We studied two different architectures of the DNA circuits. First, the free-floating circuit, or the well-mixed chemical system. In this type of system, DNA strands that are the objects of interest in our design, move freely in a liquid solution within a test tube, like almost all chemical solutions in wet lab experiments. Second, the spatially localized circuit. This architecture is developed recently to offer new advantages for DNA-based circuit, such as sped-up reaction, lower error rate, and so on. The main idea is to tether the DNA strands participating in the computational process onto a static structure, such as DNA origami; so that the computation can be restricted spatially.

We designed decision-making and reinforcement learning strategies for both architectures by taking inspiration from swarm intelligence in nature. Specifically, we worked on Immune Network Theory (for the well-mixed chemical system), and Ant Colony System (for spatially localized system). We develop models of computation that exploit the distributed interaction among participating individuals, the indirect communication method mediated by the environment, and the positive-negative feedback arise from the massively interacting network.

In this work, we also employ a decision making or action taking scenario as a test-bed for our computational model. One of the challenges in promoting a higher level of intelligence to a molecular robot is to introduce the capability to cope with the changes within its environment. Decision making is an important feature that deals with the adaptability when a robot is not equipped with complete information. In this research, DNA Strand Displacement is chosen as the main framework to develop a DNA-based decision making system. The implementation of a action taking model itself is inspired by the principle of the human body's immune system, which relies on a dstributive interaction between the involved components (referred to as antigen and antibody), which compete to remain as the best population. In this scenario, a robot is given numerous choices of actions in order to deal with a particular problem. The robot is then reinforced to decide which action is the best for any given problem.

Lastly, the implementation of DNA Strand Displacement system based-on nature-inspired computation is observed. By using the Immune Network Theory and Chemical Reaction Network, the compilation of DNA-based operation is designed and the formulation of its mathematical model is derived. Furthermore, the implementation on this system is compared with the traditional implementation by using silicon-based programming. The objective of this research is too see a positive correlation between both. Thus, we intend to seek a novel compilation method from any existed mathematical model into biochemical reaction in order to solve particular computational problem. In summary, the execution of all works in this thesis and how they are connected each other is captured by the flowchart in Figure 1.6 and the research position in Figure 1.7.

1.4 Organization

This thesis is a summary of four independent research papers on computational design of synthetic bio-molecular systems based on DNA strand displacement reaction. They may be seen as independent results; however still in a related

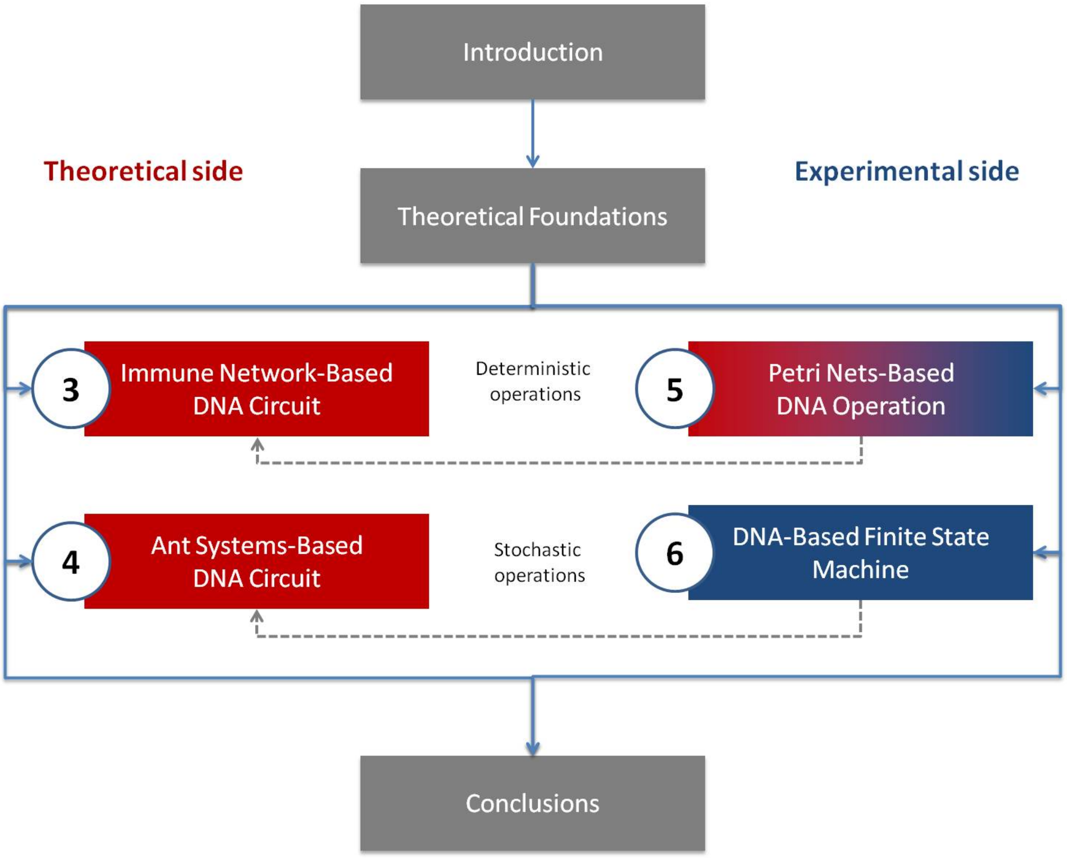


Figure 1.6: Research flowchart

scope. We will begin with some brief explanation in Chapter 2 about basic theories and frameworks that are used throughout the article. The major results of our works are reported in four different chapters, respectively from Chapter 3 to Chapter 6. The arrangement of the contents is mainly according to the time-flow of the research execution (and related papers being published); with some adjustment are made to create a coherent narrative. In Chapter 7, we finally summarize our works and point out several challenges that can be considered in the future. One additional chapter is added to present the in silico implementation of the designed systems, which may also be a useful reference.

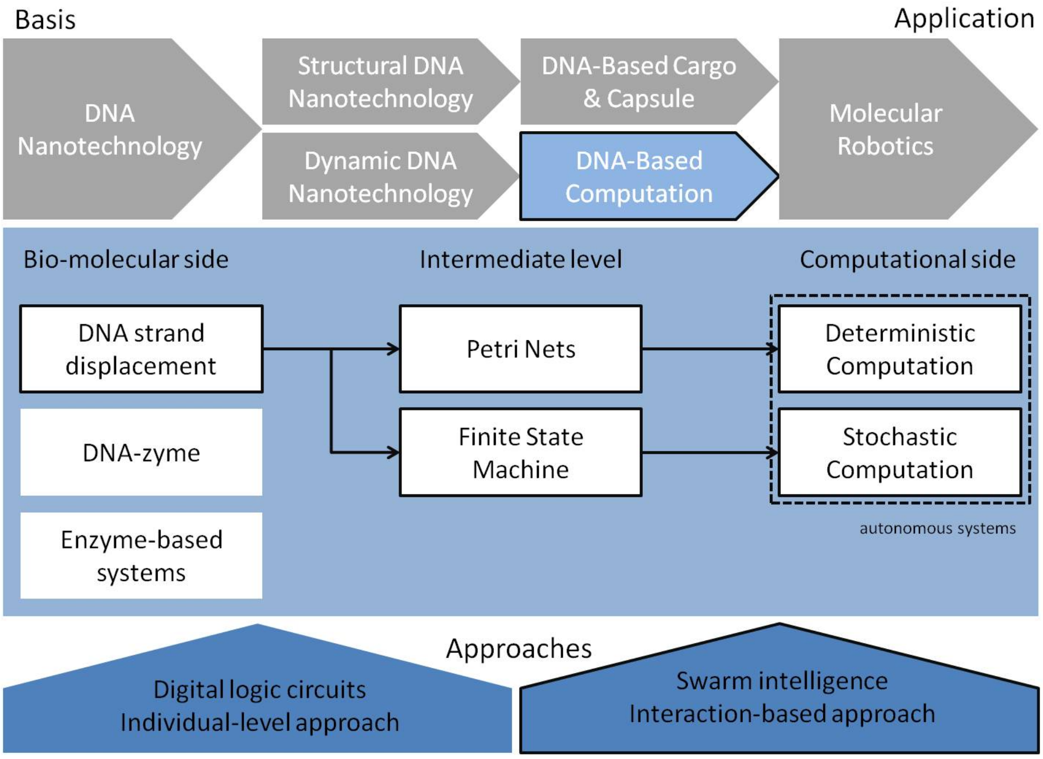


Figure 1.7: Research position, with bold-line border boxes represent what are covered in this thesis

2

Frameworks

2.1 Theoretical Foundation

This section briefly introduces the theoretical foundations used in the rest of the thesis. All other theories specific to the work of particular chapters will be discussed separately.

2.1.1 DNA Strand Displacement

Toehold-mediated three-way branch migration DNA strand displacement (or simply DNA strand displacement) is a stochastic biochemical mechanism, where single strand DNA as input signals react to pre-hybridized multi strand complexes (or gates) to release other single strand DNA as output signals through a process termed branch-migration (1). This process can be viewed as a computational mechanism, with DNA as a medium to carry the structure as well as the information. Among all DNA manipulation techniques available so far, DNA strand displacement is said to be superior as it does not require any different molecules design such as the restriction enzyme. Therefore, it is suitable for large application designs (13, 16).

Figure 2.1 shows the basic reaction of DNA strand displacement. Abstractly, a DNA strand is seen as consecutive sub-sequences (or called domains). The design of DNA sequences is done at the domain level, and a complete sequence is obtained by combining all corresponding units. By doing this, we can avoid the need to work with the nucleotide sequences directly. Instead, we treat the domain as the simplest functional unit in the computation. Arbitrary alphanumeric characters represent the coding sequence and asterisk signs represent their complements (A with T, C with G, and vice versa). One character consistently represents the exact sequence.

There are two different types of domain, depending on the sequence length. A toehold (color-coded) is a short part (between 4-6 nucleotides) that triggers the whole reaction through reversible hybridization. A non-toehold is a longer part (approximately 20 nucleotides) that provides irreversible binding power once hybridized. The reaction begins as soon as there are free toeholds in an input strand and a gate complex (domain 2 of A and 2* of T_1). Their interaction accelerates the branch migration process if the adjacent domain also shares the same region (domain 3 of A). The old domain (domain 3 of B) will be ejected, leaving the unstable binding of the other toehold domain (domain 4 of B). As a result, an output strand will be released. Note that this reaction can be reversed as the whole process exposes new toeholds that may bind each other (i.e. another reaction that consumes strand B and complex {A, C}, and produces strand A and complex {B, C}).

$$\mathrm{A} + \mathrm{T}_1 \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \mathrm{T}_2 + \mathrm{B} \tag{2.1}$$

Equation 2.1 shows the equivalent chemical reaction to the DNA strand displacement process in Figure 2.1, assuming all the sequences matches the coding and the branch migration happens instantaneously.

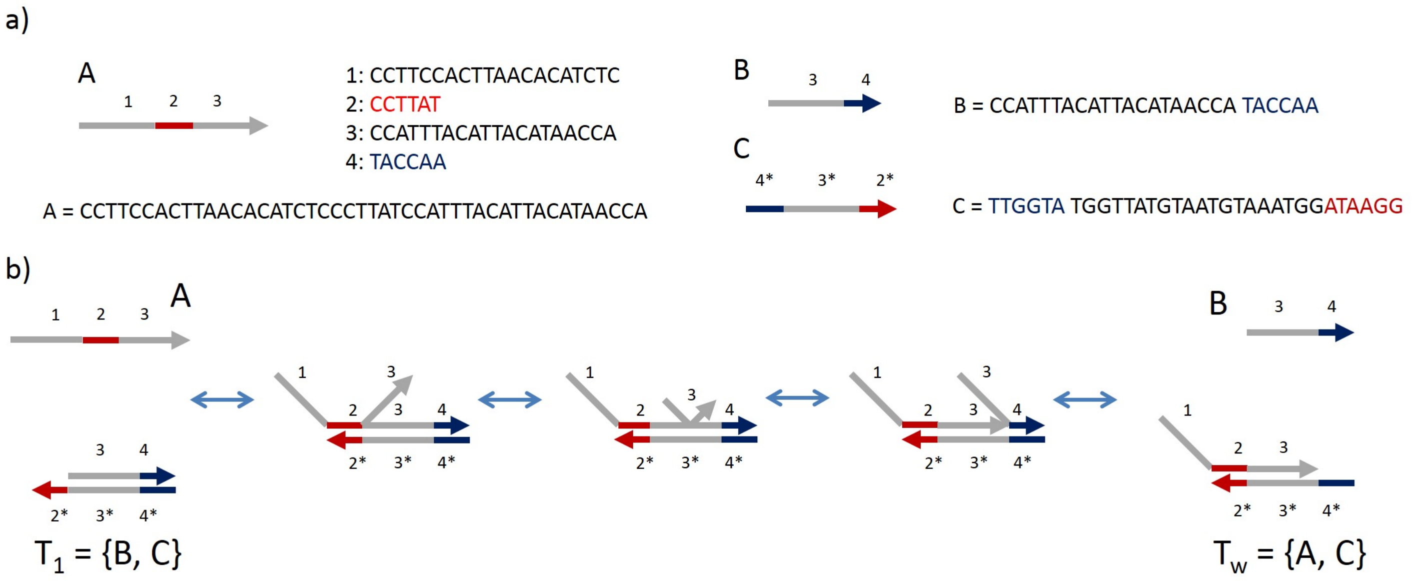


Figure 2.1: Basic of DNA strand displacement reaction

2.1.2 Chemical Reaction Networks

Chemical Reaction Networks (or CRNs) is an applied mathematics theory that models behavior of biochemical reaction systems in term of ordinary differential equation. Suppose there are chemical species A, B, and C involving in a chemical reaction as in Equation 2.2.

$$\mathrm{A} + \mathrm{B} \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \mathrm{C} \tag{2.2}$$

The kinetics of the reaction can be observed by instantaneous changes of each species as in Equation 2.3.

$$\begin{aligned} \frac{d[A]}{dt} = \frac{d[B]}{dt} &= -k_1[A][B] + k_2[C] \\ \frac{d[C]}{dt} &= k_1[A][B] - k_2[C] \end{aligned} \tag{2.3}$$

Where [A], [B], and [C] are population numbers of species at any time t. The reaction can be divided into two different reactions. First, a reaction where species A and B act as the reactants, and C as the product (or the forward reaction with constant rate k_1). Second, a reaction where species C acts as the reactant, and A and B as the products (or the backward reaction with constant rate k_2). The number of products depends on how many reactants' molecules collide each other through the reaction, which is denoted by the reaction rate.

Therefore, in the case of the forward reaction, it is shown by the first line of Equation 2.3. The second line depicts the backward reaction. The plus and minus signs distinguish the products from the reactants, which should be positive when the species are produced and negative if they are consumed.

Soloveichik, et al. in (10) showed that CRNs can be utilized as a formulating tool in programming arbitrary DNA-based systems with complex behavior. Since DNA is basically a biochemical substrate, its reaction kinetics also follows the CRNs principle. By seeing this as a reverse problem, corresponding DNA operations can be designed to achieve the intended behavior of chemical systems. For example, in (10) a DNA-based Lotka-Volterra model and a limit cycle oscillator are successfully implemented.

2.2 Methods

We evaluate our design by means of wet-lab experiments and software-based simulations, as presented in this section.

2.2.1 Wet-Lab Experiments

All sequence designs were performed with computer-assistance at the domain level. First, a set of nucleotide sequences were generated, according to the following rules: 1) one sequence is made from only three bases: A, C, and T to minimize secondary structure formation (G only occurs in complementary sequences), 2) no more than four consecutive As and Ts or three consecutives C are included in a row to reduce synthesis error, and 3) the number of Cs is maintained in between 30%-70% to ensure comparable melting temperature (13) (52). Second, after generation, all sequences were checked by using NUPACK (58) to simulate the secondary structure and interaction, and were manually modified if necessary.

After design, DNA oligonucleotides were purchased from Eurofins Genomics, unpurified or purified by high performance liquid chromatography (HPLC) in powder form. To prepare master-stock solutions, these DNA oligonucleotides were suspended in distilled water (Millipore) to reach a concentration of 100 μM. From each stock, experiment-ready solutions were made, by first diluting the master stocks to 20 μM with 1x TAE/Mg^{2+} (containing 12.5 mM magnesium acetate) buffer, and then quantifying the correct concentration by UV absorbance at 230 nm. This quantification was performed by using a Malcom Micro UV-VIS Spectrophotometer (ES-2). Each sample was measured at least twice and the values were averaged to confirm the correctness of the quantification. From the measured results, extra 1x TAE/Mg^{2+} buffer was added to achieve stoichiometry-correct 10 μM stock solutions.

DNA gates were annealed together by mixing 6 μl of all component strands from experiment-ready stocks to make 50 μl solutions. For sequential gates, this means adding 32 μl of 1x TAE/Mg^{2+} buffer; and for synchronization and concurrent gates, this means adding 26 μl of 1x TAE/Mg^{2+} buffer to the mixed solution. Annealing process was performed by using a Bio RAD C1000 Touch

Thermal Cycler, by first heating the solutions to 95^{o}C for 10 minutes and then slowly cooling to 20^{o}C at a rate of -1^oC/minute.

Kinetics experiments were performed by using a Hitachi F7000 spectrophotometer with an Eyela NCB-1200 thermal bath. Annealed gate samples were mixed together with input signals in a 0.9 ml 28-F/MS cuvette with magnetic stirring (Pacific Science Corp.) The Solutions were prepared to reach a total volume of 600 μl. The concentration of output signals was monitored by the FRET technique, by reading the fluorescence excitation changes corresponding to particular output signals (details for this method can be found in the supplementary information). The two different fluorescence molecules used in these experiments were FAM (excitation: 495 nm, emission: 520 nm) for all operator experiments, and ROX (excitation: 580 nm, emission: 610 nm) for concurrent operator experiments. As our machine could only measure one wavelength at one time, for sequential and synchronization operators, time-scan measurements were performed. However, for the concurrent operator, monitoring was performed at the FAM channel. Wavelength measurements were performed to compare the initial and final conditions of both fluorescences. All kinetics experiments were done at 25^{o}C. The following figures depict instruments used in this work.

2.2.2 Software-Based Simulations

We set up software-based simulations to evaluate the DNA-based implementation in-silico by using Visual DNA Strand Displacement Simulator from Microsoft (http://boson.research.microsoft.com/webdna/).

This tool provides a programming language for designing composable DNA circuits based on toehold exchange, branch migration, and strand displacement as the main computational mechanisms. It compiles a collection of DNA strands into a reaction network, and provides reaction kinetics simulation based on deterministic or stochastic algorithms (59). Visual DSD has been used to verify many actual implementation of DNA-based systems, which motivated the use of this software in our study (53) (52).

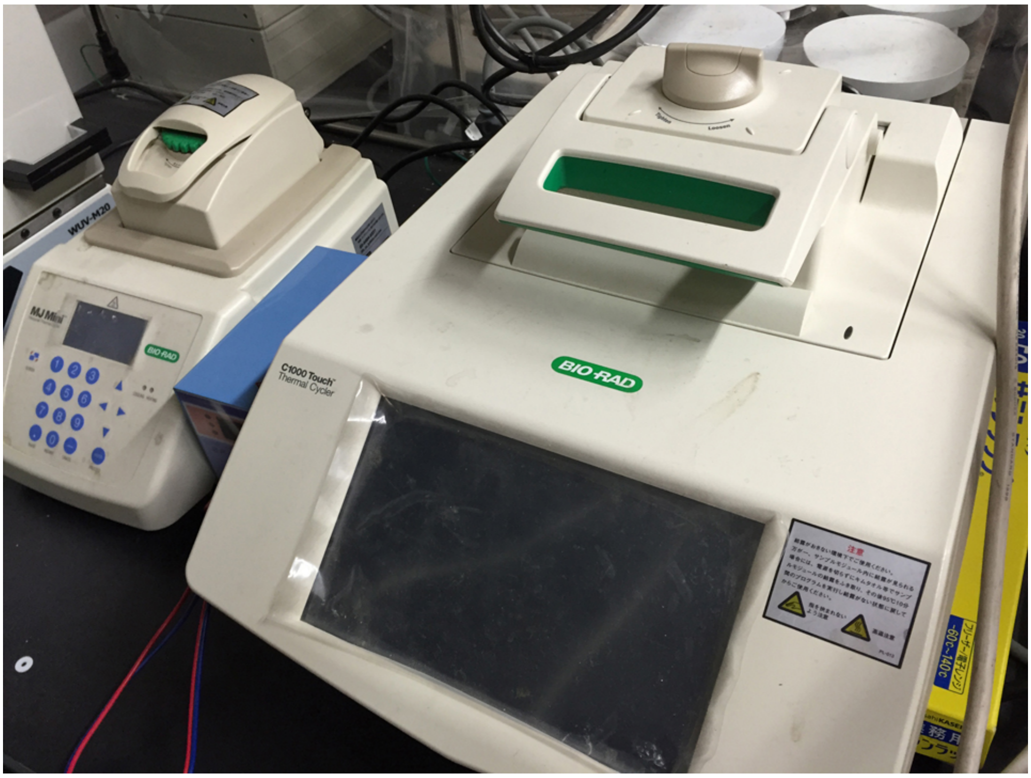


Figure 2.2: Thermal-cycler machines

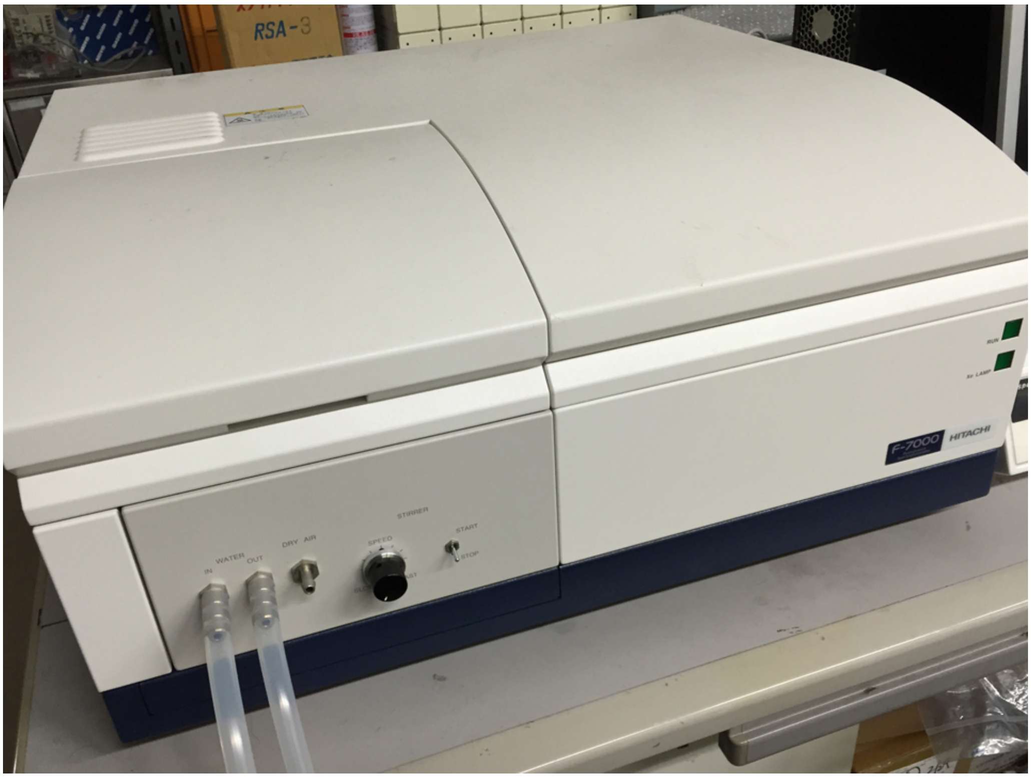


Figure 2.3: Spectrophotometer for fluorescence measurement

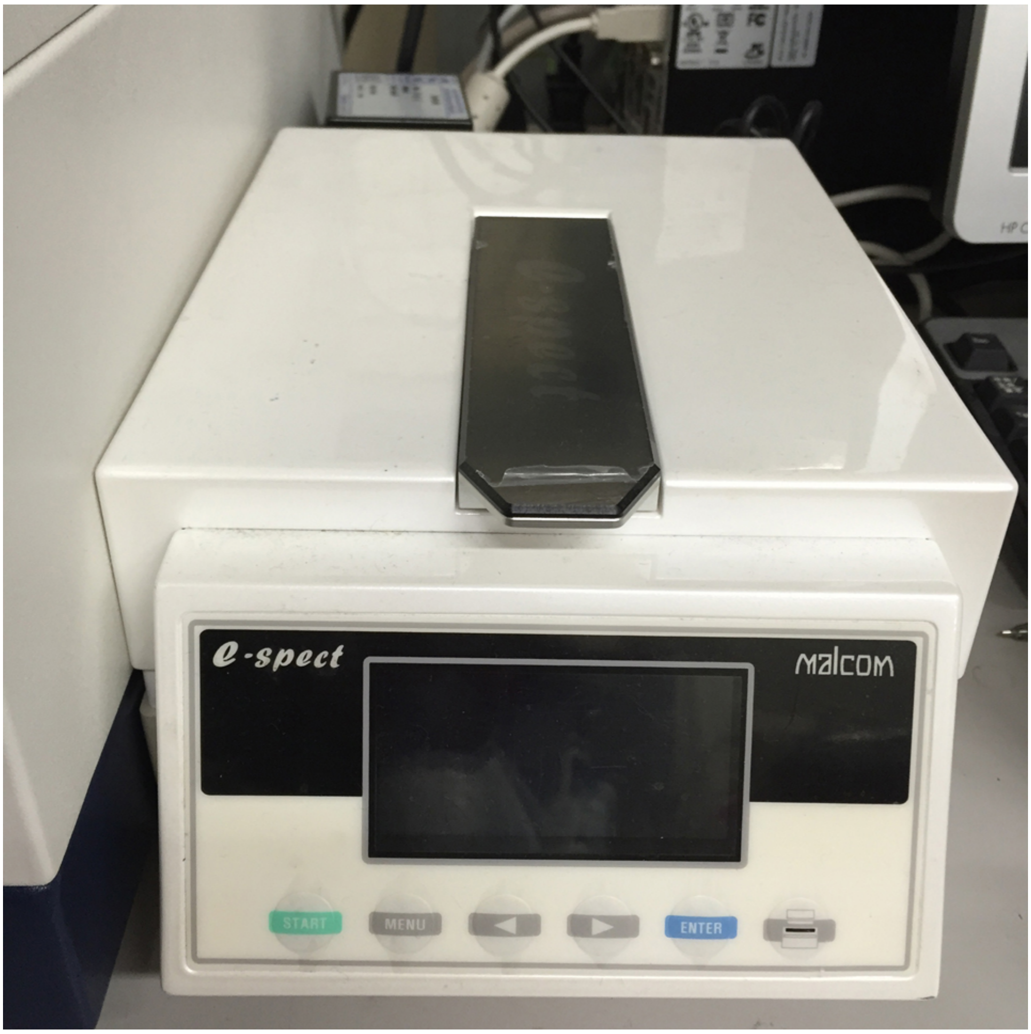


Figure 2.4: UV spectrometer for DNA quantification

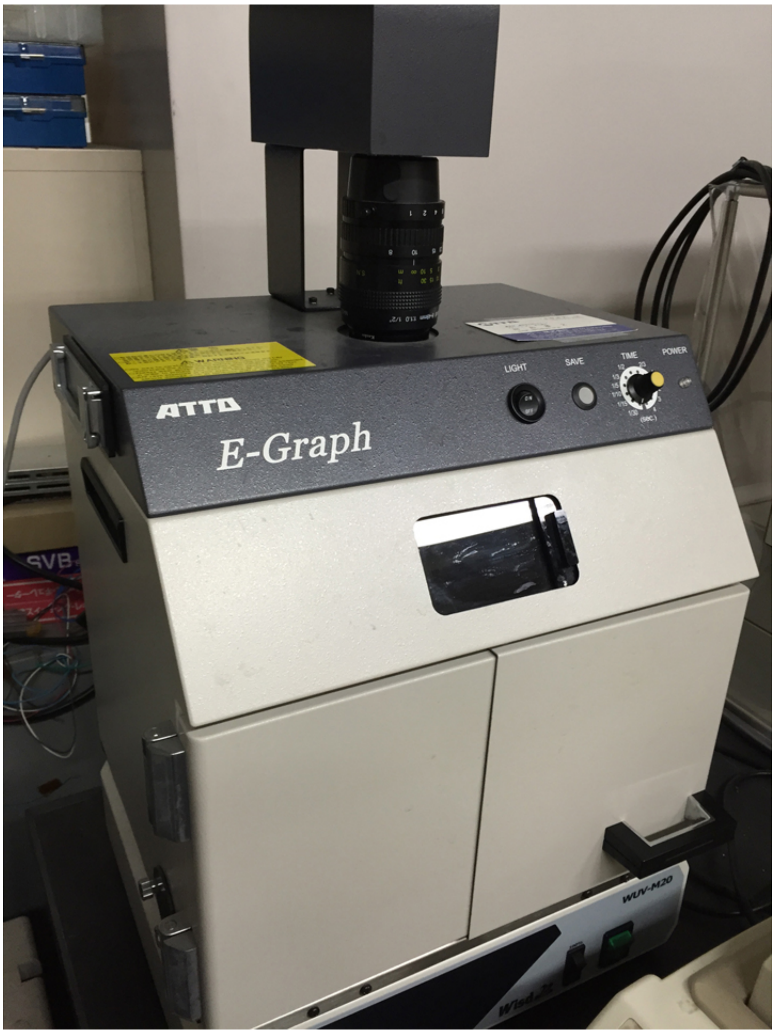
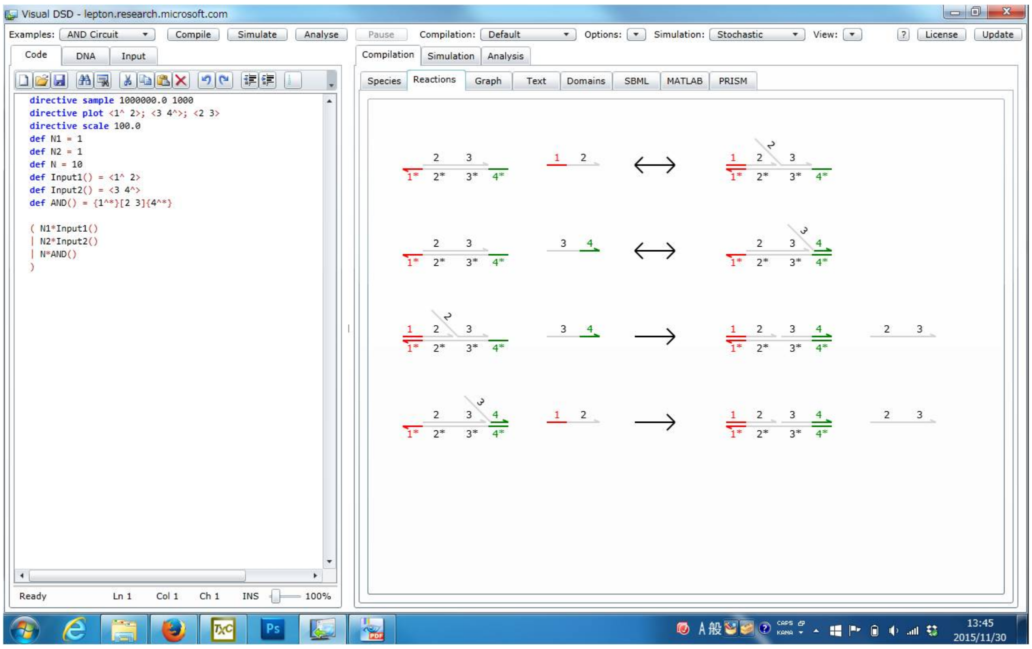


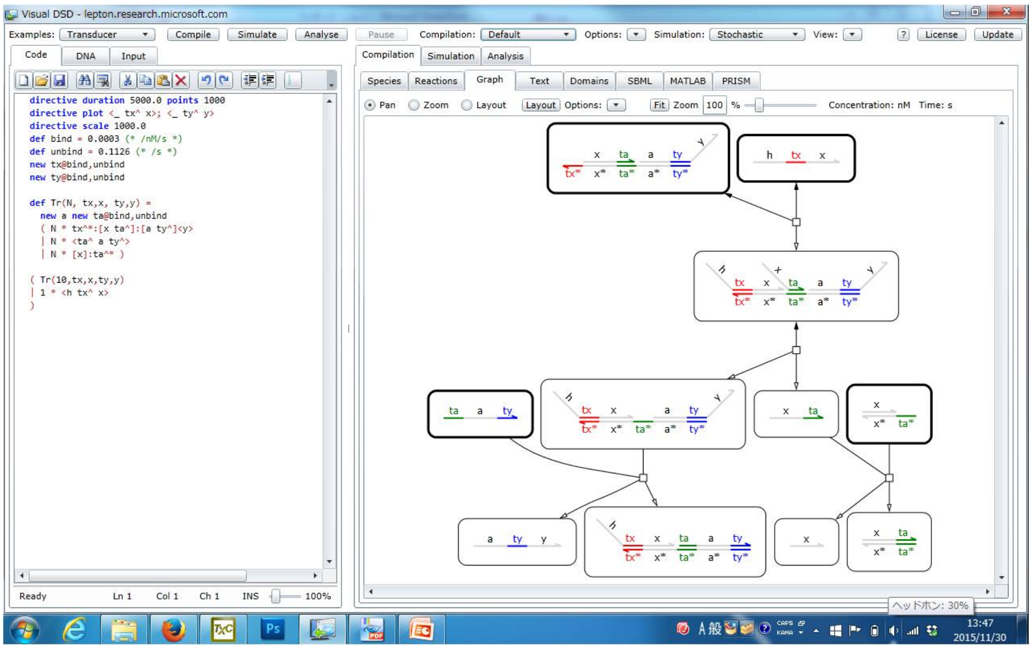
Figure 2.5: Camera for gel imaging



```
directive sample 1000000.0 1000
directive plot <1^ 2>; <3 4^>; <2 3>
directive scale 100.0
def N1 = 1
def N2 = 1
def N = 10
def Input1() = <1^ 2>
def Input2() = <3 4^>
def AND() = {1^*}[2 3]{4^*}

( N1*Input1()
| N2*Input2()
| N*AND()
)
```

Figure 2.6: Screen-shoot of Visual DSD code compilation

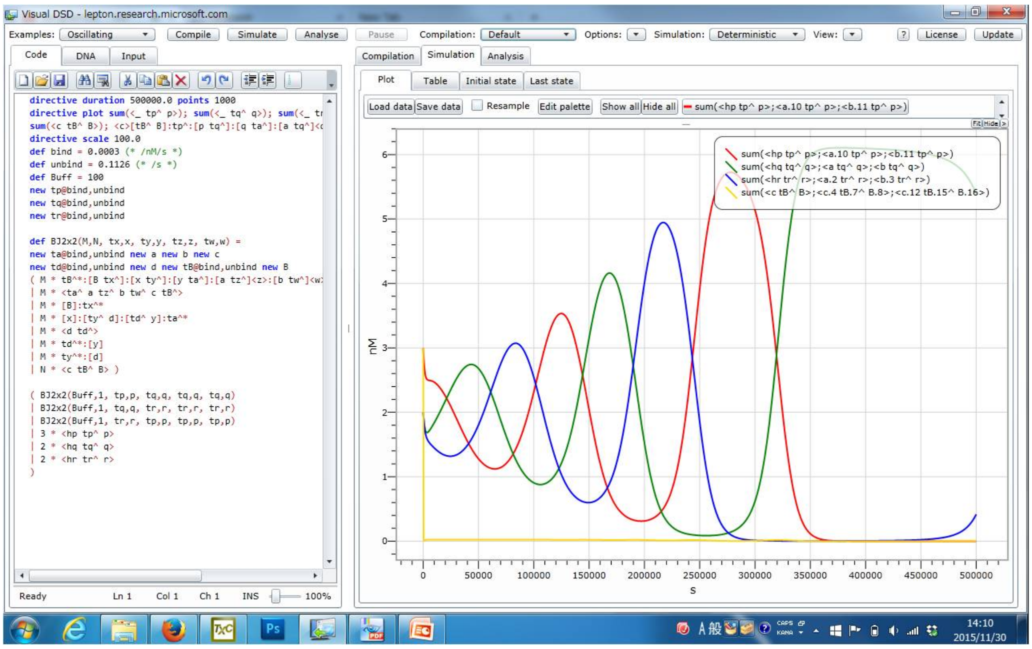


```
directive duration 5000.0 points 1000
directive plot <_ tx^ x>; <_ ty^ y>
directive scale 1000.0
def bind = 0.0003 (* /nM/s *)
def unbind = 0.1126 (* /s *)
new tx@bind,unbind
new ty@bind,unbind

def Tr(N, tx,x, ty,y) =
  new a new ta@bind,unbind
  ( N * tx^*:[x ta^]:[a ty^]<y>
  | N * <ta^ a ty^>
  | N * [x]:ta^* )

( Tr(10,tx,x,ty,y)
| 1 * <h tx^ x>
)
```

Figure 2.7: Screen-shoot of Visual DSD code compilation (reaction network graph)



Visual DSD - lepton.research.microsoft.com

Examples: Oscillating

Compilation: Default

Simulation: Deterministic

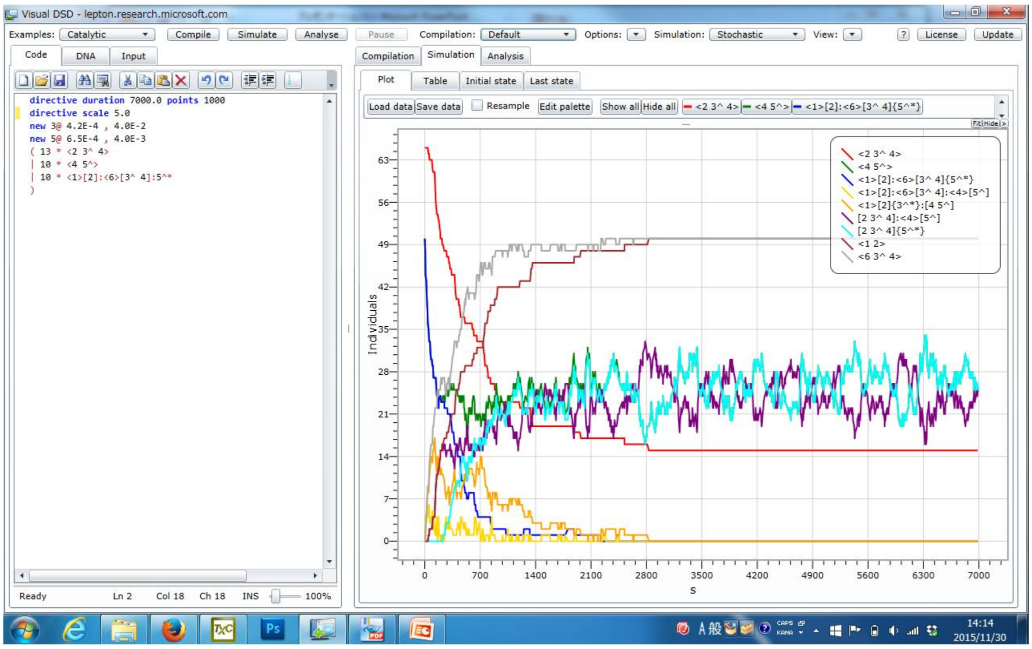
```
directive duration 500000.0 points 1000
directive plot sum(<_ tp^ p>); sum(<_ tq^ q>); sum(<_ tr
sum(<c tB^ B>); <c>[tB^ B]:tp^:[p tq^]:[q ta^]:[a tq^]<c
directive scale 100.0
def bind = 0.0003 (* /nM/s *)
def unbind = 0.1126 (* /s *)
def Buff = 100
new tp@bind,unbind
new tq@bind,unbind
new tr@bind,unbind

def BJ2x2(M,N, tx,x, ty,y, tz,z, tw,w) =
new ta@bind,unbind new a new b new c
new td@bind,unbind new d new tB@bind,unbind new B
( M * tB^*:[B tx^]:[x ty^]:[y ta^]:[a tz^]<z>:[b tw^]<w
| M * <ta^ a tz^ b tw^ c tB^>
| M * [B]:tx^*
| M * [x]:[ty^ d]:[td^ y]:ta^*
| M * <d td^>
| M * td^*:[y]
| M * ty^*:[d]
| N * <c tB^ B> )

( BJ2x2(Buff,1, tp,p, tq,q, tq,q, tq,q)
| BJ2x2(Buff,1, tq,q, tr,r, tr,r, tr,r)
| BJ2x2(Buff,1, tr,r, tp,p, tp,p, tp,p)
| 3 * <hp tp^ p>
| 2 * <hq tq^ q>
| 2 * <hr tr^ r>
)
```

Ready Ln 1 Col 1 Ch 1 INS 100%

Figure 2.8: Screen-shoot of deterministic simulation



Visual DSD - lepton.research.microsoft.com

Examples: Catalytic

Compilation: Default

Simulation: Stochastic

```
directive duration 7000.0 points 1000
directive scale 5.0
new 3@ 4.2E-4 , 4.0E-2
new 5@ 6.5E-4 , 4.0E-3
( 13 * <2 3^ 4>
| 10 * <4 5^>
| 10 * <1>[2]:<6>[3^ 4]:5^*
)
```

Ready Ln 2 Col 18 Ch 18 INS 100%

Figure 2.9: Screen-shoot of stochastic simulation

3

Immune Network-Based DNA Circuit

One of the biggest obstacles in the molecular programming is that there is still no direct method to compile arbitrary mathematical models into biochemical reactions in order to solve given computational problems. In this paper, the application of the DNA strand displacement system based on the nature-inspired computation is observed. By using the Immune Network Theory and Chemical Reaction Networks, a compilation of DNA-based operationsand the formulation of its mathematical model are derived. Furthermore, the application of this system is compared with the conventional implementation by silicon-based programming. From results obtained, we can see a positive correlation between the two. One of the possible applications of this DNA-based model is for a decision-making scheme of intelligent computers and molecular robots.

3.1 Introduction

Both theoretical studies and experimental results have been demonstrated on DNA nanomachine devices (5, 7, 10, 12, 13, 51, 52). A toehold mediated strand displacement and branch migration mechanism have been treated as a promising methodology to provide dynamical systems, which allows the bio-chemical circuits to approach the silicon-based machines functionality. While most attention has been paid to the development of the mechanical function of nucleic-acid based molecular agents, such as DNA walkers and motors (9, 18, 19, 20, 21, 22), there is still a large gap where there is not many references can be found regarding the design of computation and information processing strategies by using DNA.

One of the biggest drawbacks of DNA-based information processing system is that performing computation on the DNA strands is a non-trivial task. Even nucleotides can store information in a manner similar to binary numbers, the possible manipulation is limited. This makes the coding of mathematical operations in bio-computers is not as straightforward as in dry computers. On the contrary, the predictable behavior of the Watson-Crick pairs and simple hybridization can be used as a mechanism to program at the molecular level (10, 13, 16, 17, 60). In (52) and (61), nature inspired computational methods, such as neural-networks and the artificial immune system, have been used as strategies to develop DNA-based systems capable of decision-making. It opens the possibility of DNA to perform more complex tasks in various intelligent applications. However, the design of DNA reactions and motifs are still limited to some specific problems. This leaves a question: given an arbitrary mathematical model to solve a computational problem, can we directly build a DNA-based system with the same behavior?

In this work, we extend the idea of a self-organizing algorithm, namely Immune Network Theory, to describe a DNA-based interaction system. This model has been used to solve many computational problems, including decision-making and reinforcement learning (62, 63) in machine learning, robotics, and artificial intelligence. Here, we obtain the correlation between the DNA-based implementation with the mathematical formulation, in order to seek a more schematic way to compile DNA Strand Displacement reactions.

To approach the solution to this problem, we express the DNA reactions above mathematical notation referred to as Chemical Reaction Networks (CRNs). For decades, CRNs has been used to describe and analyze the kinetics of chemical reaction systems. Given arbitrary chemical reactions, in which substrate of reactants are well-mixed in a defined reaction rate constant to produce other chemical products, we can obtain instantaneous changes of substrate population from time to time, in terms of differential equations. The CRNs has also been demonstrated as a powerful tool for molecular programming because it can simulate a variety of complex dynamical systems implemented by DNA Strand Displacement reactions, including the Lotka-Voltera model (10).

In general, our contribution can be divided into two. First, we develop a new DNA computing algorithm that is based on the principles of swarm intelligence and interaction between DNA strands in vitro. Second, we bridge the current DNA computing systems to the existing mathematical model, so that in the future the gap between traditional and molecular computation can be minimized.

3.2 DSD-Based Computational Operator

The compilation of computational procedures from DNA strand displacement reactions can be done in various ways. Among all available design to date, the fundamental is to encode information signals as single strand DNA with uniform structures. For example, in (60, 64) the 3-domains design was introduced. The work in (10) employed the 4-domains structure instead. Additionally, a set of multi-strand DNA complexes or gates are treated as the fuel for the reactions. Their design varies depending on input and output signals. To maintain the reactions run long enough, we assume that the amount of the multi-strand DNA are largely available in the system that it will not exhaust within the given time.

In this chapter, our DSD-based computational operators (or simply DNA operator) are implemented by following the principles as outlined in (10). We employ the 4-domains coding for DNA signals, which consists of two non-toehold domains and two toehold domains situated between each other. The first non-toehold is the history domain. It stores information of previous interaction, to which DNA gates the signal bound previously. Two same signals may possess two different history domains, as they may be produced from different reactions. The first toehold is the reaction initiator. It triggers the whole branch migration and strand displacement processes. The second non-toehold is the signal identity. It is unique for each single strand DNA as it captures different information. The second toehold is an additional domain that is necessary to establish irreversible reaction in particular steps.

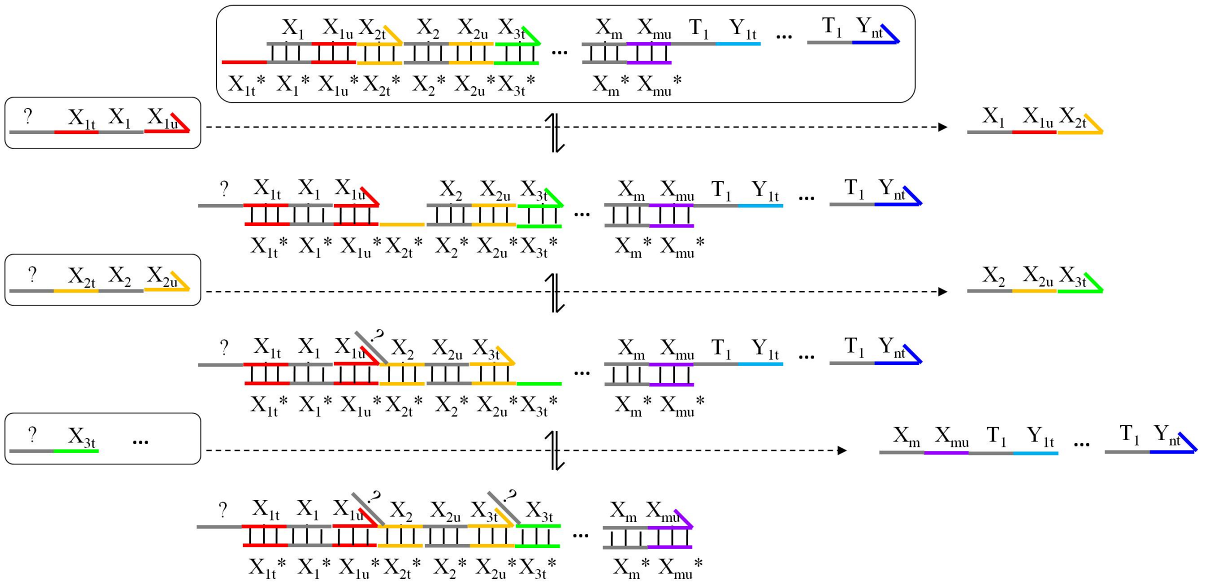

?
X_{1t} X_1 X_{1u}
X_1 X_{1u} X_{2t} X_2 X_{2u} X_{3t} ... X_m X_{mu} T_1 Y_{1t} ... T_1 Y_{nt}
X_{1t}* X_1* X_{1u}*X_{2t}* X_2* X_{2u}*X_{3t}* X_m* X_{mu}*
X_1 X_{1u} X_{2t}
X_{2t} X_2 X_{2u}
X_2 X_{2u} X_{3t}
X_{3t} ...
X_m X_{mu} T_1 Y_{1t} ... T_1 Y_{nt}

Figure 3.1: General DNA operator - input binding part

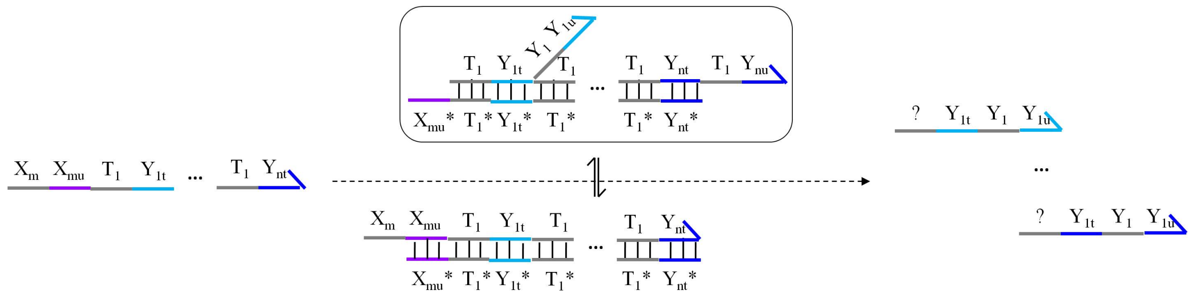

X_m X_{mu} T_1 Y_{1t} ... T_1 Y_{nt}
T_1 Y_{1t} Y_1 Y_{1u} T_1 ... T_1 Y_{nt} T_1 Y_{nu}
X_{mu}* T_1* Y_{1t}* T_1* T_1* Y_{nt}*
X_m X_{mu} T_1 Y_{1t} T_1 ... T_1 Y_{nt}
? Y_{1t} Y_1 Y_{1u}
...
? Y_{1t} Y_1 Y_{1u}

Figure 3.2: General DNA operator - output releasing part

A template of general-purpose DNA gates can be designed that accepts m number of input strands to produce n number of output strands (m and n are arbitrary positive integers). There are two main parts of this operator. The first part is the structure that receives all input strands, or referred to as the binding-operator, which is shown in Figure 3.1. When an input strand (left-side) binds to the binding-operator (center), an auxiliary strand (a single strand with a different structure with the input signal) is released (right-side) as denoted by the horizontal dashed arrow. The structure of the binding-operator is changed and a new toehold domain for the following reaction becomes available (denoted by the vertical reversible arrow). Then, the next input signal may proceed consecutively until the last expected input presents (the recurrent process until the m-th iteration is shown by the ellipsis mark). Eventually, a new longer intermediate single strand DNA, as referred to as the trigger strand, is released. This step marks the binding of all inputs and the trigger strand will be used in the next part of the DNA operator to signal the release all output strands as shown in Figure 3.2.

The second part is the structure that releases all output strands, or referred to as the releasing-operator. It has a simpler structure compared to the first one. The length of the trigger strand depends on the number of the output strands. It consists of a toehold that has a free complement available in the releasing operator. As these toeholds bind, the rest of the sequences thereafter displaces all the single strand DNA bound to this complex, which will be the final output strands. This last step depicts the transformation of m number of input strands into n number of output strands. This is sufficient to our objective to design the event-based computation by DNA. The number of single strand DNA in the system remains constant, whether in a form of active signals or bound to DNA gates. On the contrary, the active multi-strand DNA complexes will be turned into wastes as soon as they are involved in a reaction. All DNA strands that have to be initially prepared are denoted by the rectangular shapes.

$$\begin{aligned} \mathrm{X_1 + Ti_1} &\underset{k_2}{\overset{k_1}{\rightleftharpoons}} \mathrm{Ti_2 + I_1} \\ &\dots \\ \mathrm{X_{m-1} + Ti_{m-1}} &\underset{k_y}{\overset{k_x}{\rightleftharpoons}} \mathrm{Ti_m + I_{m-1}} \\ \mathrm{X_m + Ti_m} &\xrightarrow{k_i} \mathrm{Ti_{waste} + Trig} \end{aligned} \tag{3.1}$$

$$\mathrm{Trig + To} \xrightarrow{k_j} \mathrm{To_{waste} + Y_1 + \dots + Y_n} \tag{3.2}$$

This whole operation is equivalent to a chemical system, as denoted by Equation 3.1 and Equation 3.2, where $X_1, ..., X_m$ are a set of input strands, $I_1, ..., I_{m-1}$ are a set of intermediate strands, and $Y_1, ..., Y_n$ are a set of output strands respectively; $Ti_1, ..., Ti_m$, To, Ti_{waste} and To_{waste} are sets of multi strand DNA complexes or DNA gates; $Trig$ is the trigger strand; and k_1, ..., k_i, k_j captures the reaction rates.

$$\mathrm{X_1 + \dots + X_m} \xrightarrow{k_i} \mathrm{Y_1 + \dots + Y_n} \tag{3.3}$$

From the point of view of computation, these reactions can be simplified into a transformation function that converts the m-number of input signals into n-number of output signals, as depicted by Equation 3.3. All auxiliary strands other than input and output signals are semantically not important in this case (note that from the point of view of experiments, they may affect the kinetics difference). To make their presents insignificant, we assume that their amount can be maintained constant throughout the reaction, either by providing a very large number of population or constantly replenishing the depleted fuels. These auxiliary strands are also treated as inputs with amount a lot bigger than the DNA gates, since the increasing of their numbers are linear the consumed DNA complexes.

3.3 DNA Mathematical Model

3.3.1 Jerne's Immune Network Model

Living organism is an indefinitely huge system consisting of small billion particles working together to provide several functions. One of the example is the human's natural immune system. Whenever our body encounter the present of foreign molecules, referred to as antigen, it triggers a self-defense mechanism to cancel the attack. While it is still unclear how this multilayer system works specifically, it is known that this defense mechanism is solely driven by distributed interaction of specific part of the immune system's cells, referred to as antibody.

For most of the cases, the immune system can respond accordingly, despite the huge numbers and unlimited possibilities of the antigen types, without given any prior information. This may happen through the recognition process between the antibody and the antigen, which principle is similar to the key-and-lock mechanism. Both antibodies and antigens are "made" of certain receptor combinations that match each other. When they find their suitable matching, an antibody binds and kills the antigen through further processes that involve a different type of cells, which also stimulate the production of the antibody from the same type. As their number increases, the more antigens will be bound and eliminated. On the contrary, the rest type of the antibodies naturally die in the absence of interaction with the antigen. This competing process between the positive and negative feedback leaves only the fittest antibody to work in the system effectively. This shows the adaptability of the natural immune system (63), which can be adopted in designing a learning model and decision-making capability of the DNA-based system.

Furthermore, not only the immune system can respond correctly to the initially unknown stimulus, it also responds faster in the second occurrence of the similar pathogen attack. While how this mechanism works is yet still unrevealed, some studies mentioned that this occur due the "memory" as well as the learning capability of the immune system. One of the most popular hypothesis, termed the Immune Network Theory (proposed by Niels Jerne (62)), stated that the interaction within the immune system does not only happen between the antibodies

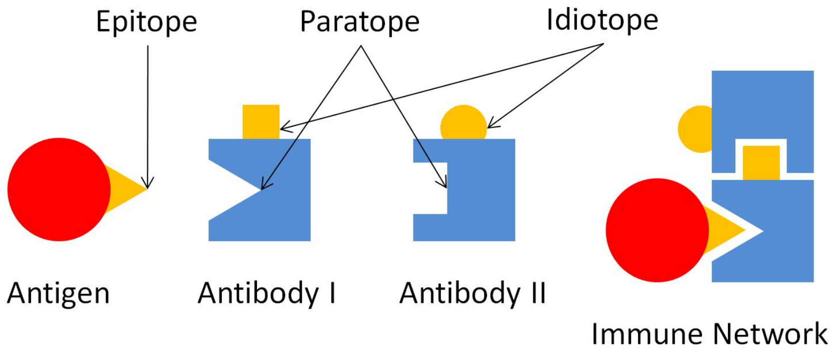


Figure 3.3: Model of immune network

and antigens, but also among the antibodies themselves. As the results, they maintain the number of each population from the past experiences in relatively a high amount. As a new and different antigen comes attacking, the balance of the system will be disrupted. They again converges into another new stable state if the better new antibody is found.

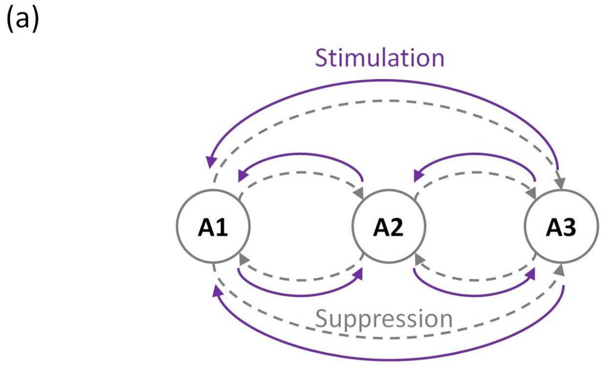
This interaction can be abstractly represented as building blocks, which is shown in Figure 3.3. Suppose the different types of antigens and antibodies are shown by particular shapes, and the matching between them are captured by those are complementary to each other. In the immune system, these parts are termed as an epitope for the antigen and a paratope for the antibody. The binding process can happen when a paratope locks an epitope. In this case, the antibody is said stimulated and the antigen is said suppressed. For example, the antigen's epitope in Figure 3.3 matches with antibody I's paratope, but not with antibody II's one. However, according to Jerne's theory, instead of only having a paratope as a binding side, an antibody also possesses another region called an idiotope which binds to other paratopes. Therefore, an antibody with an idiotope is treated similarly to an antigen and it will be suppressed by another antibody with a paratope. From Figure 3.3, this is captured by the relation of antibody I and antibody II. According to these interactions, there are indirect communication mechanisms between antibodies maybe be established at the large-scale. They provide indirect response to the previously unknown molecule's attack.

$$\begin{aligned}
\frac{d[a_i(t)]}{dt} &= \left[\sum_{j=1}^{M} m_{ij}a_j(t) - \sum_{k=1}^{M} m_{ki}a_k(t) + \sum_{n=1}^{N} m_{in}b_n(t) - k_i\right] a_i(t) \quad (3.4) \\
\frac{d[b_n(t)]}{dt} &= -\sum_{n=1}^{N} m_{ni}a_i(t)b_n(t) \quad (3.5)
\end{aligned}$$

The mathematical formulations of the Immune Network Theory for antibody a_i and antigen b_n at time t are denoted by Equation 3.4 and 3.5. Suppose there are M-number of antibodies and N-number of antigen, there are four important terms to describe the changes of an antibody through time: 1) the stimulation rate with other antibodies to which its paratope binds, 2) the suppression rate with other antibodies to which its idiotope is bound, 3) the stimulation rate with antigens, and 4) the natural death rate. The affinity is a parameter that describes the closeness between antibody i and antibody j, which is described by parameter m_{ij}. This determines the quantitative degree of how they match each other (1 means a perfect match, and 0 means totally different). The affinity between antibody i and antigen b_n is denoted by m_{in}, and the natural death's rate of the antibody a_i is depicted by k_i. On the other hand, the change of the antigen is only determined from its interaction to antibodies, which is denoted by parameter m_{ni}. W discuss how to compile this equation into DNA reactions in the following section.

3.3.2 Formalization of DNA-Based System

In compiling Immune Network Theory into DNA reactions, the first thing to do is to determine the structure of the interaction network itself. Suppose there are three arbitrary and different antibodies: A_1, A_2 ,and A_3 interacting in an Immune Network model as Figure 3.4.a. The stimulation rates are shown by solid colored lines, while the suppression rates are shown by dashed gray lines. The antibodies and antigens interaction are represented by this stimulation and suppression rates, which are represented by affinity values ranging from 0 to 1. These values also show how fast an antibody stimulates or suppresses others. Therefore, the affinity values are similar to the reaction rates in a chemical reaction system. Figure 3.4.b



(b)

Stimulation matrix

	A1	A2	A3
A1		α_{12}	α_{13}
A2	α_{21}		α_{23}
A3	α_{31}	α_{32}	

Suppression matrix

	A1	A2	A3
A1		β_{12}	β_{13}
A2	β_{21}		β_{23}
A3	β_{31}	β_{32}	

Figure 3.4: Interaction model between antibodies and affinity matrices

shows the derivation of the affinity matrix that determines the affinity values of the antibodies (note that affinities values with antigens are not shown in this figure).

The affinity values are reciprocal. Therefore, the relation between the stimulation and suppression rates for particular antibody x and y satisfies $\alpha_{xy} = c \times \beta_{xy}$, for any constant c. The ratio between these rates varies depending on particular antibodies (e.g. the ratio between α_{12} and β_{12}; and α_{13} and β_{13} may not be the same). For simplification reason, we assume that this ratio is uniform for all antibodies. Therefore, the suppression matrix is actually a scalar multiplication of the stimulation matrix. In our model, we use $c = 1$, so that $\alpha_{12} = \beta_{12} = k12$.

Lets consider an example of antibodies A_1 and A_2. A_2 is stimulated by A_1 (consequently, A_1 is suppressed by A_2). Thus, the increasing level of A_2 is equal to the decreasing level of A_1, ruled by reaction rate k_{12}. This interaction may only happen when both species exist within the system. Therefore, this is equivalent to the consumption of A_1 and the production of A_2 in the presence of both. As shown in Equation 3.6, the number of A_1 is reduced by one in the reactants side or the left-hand side of the equation, while the number of A_2 is added by one in the products side or the right-hand side of the equation. The same principle can be applied to the rest of interaction from Figure 3.4, to obtain their equivalent chemical reactions.

$$\begin{gathered}
\mathrm{A}_1 + \mathrm{A}_2 \xrightarrow{k12} \mathrm{A}_2 + \mathrm{A}_2 \\
\mathrm{A}_1 + \mathrm{A}_3 \xrightarrow{k13} \mathrm{A}_3 + \mathrm{A}_3 \\
\mathrm{A}_2 + \mathrm{A}_3 \xrightarrow{k23} \mathrm{A}_3 + \mathrm{A}_3 \\
\mathrm{A}_2 + \mathrm{A}_1 \xrightarrow{k21} \mathrm{A}_1 + \mathrm{A}_1 \\
\mathrm{A}_3 + \mathrm{A}_1 \xrightarrow{k31} \mathrm{A}_1 + \mathrm{A}_1 \\
\mathrm{A}_3 + \mathrm{A}_2 \xrightarrow{k32} \mathrm{A}_2 + \mathrm{A}_2
\end{gathered} \tag{3.6}$$

The antigen-antibody interaction, where γ is the best-match antibody that can be substituted by either A_1, A_2 or A_3; and X is the antigen, can be determined in the similar way. Equation 3.7 and 3.8 shows about this, and the death rate that limits the population growth from exploding.

$$\gamma + \mathrm{X} \xrightarrow{kx} \gamma + \gamma \tag{3.7}$$

$$\begin{aligned} \mathrm{A}_1 &\xrightarrow{d1} \emptyset \\ \mathrm{A}_2 &\xrightarrow{d2} \emptyset \\ \mathrm{A}_3 &\xrightarrow{d3} \emptyset \end{aligned} \tag{3.8}$$

The kinetics of the chemical systems can be observed from Equation 3.6 to Equation 3.8, by using CRNs as previously described in Section 2.1.2. This is as summarized by Equation 3.9 and 3.10.

$$\begin{aligned} \frac{d[A_1]}{dt} &= ((k_{12}[A_2] + k_{13}[A_3]) - (k_{21}[A_2] + k_{31}[A_3]) + k_x[X] - d_1)[A_1] \\ \frac{d[A_2]}{dt} &= ((k_{21}[A_1] + k_{23}[A_3]) - (k_{12}[A_1] + k_{32}[A_3]) + k_x[X] - d_2)[A_2] \\ \frac{d[A_3]}{dt} &= ((k_{31}[A_1] + k_{32}[A_2]) - (k_{13}[A_1] + k_{23}[A_2]) + k_x[X] - d_3)[A_3] \end{aligned} \tag{3.9}$$

$$\frac{d[X]}{dt} = -k_x[\gamma][X] \tag{3.10}$$

From here, the CRNs of our Immune Network Model can directly be compared to the Equation 3.4 and 3.5. For example, all species in Equation 3.9 have equal terms to Equation 3.4. On the first term, the affinity value m_{ij} is equal to the reaction rate, and the stimulating antibody $a_j(t)$ is equal to the different species that stimulate it. On the second term, the affinity m_{ki} is also equal to the reaction rate, and the suppressing antibody $a_k(t)$ is equal to different species that suppressing it. On the third term, the reaction rate with the antigen and its number at the given time determine the affinity value m_{in}. One the last term, it is shown the death rate of each species. Equation 3.5 and 3.10 undergo the similar analysis as they show the correlation between antigens model. From here, we can implement the chemical reactions in Equation 3.6 to Equation 3.8 into corresponding DNA operators by following the outline as explained in Section 3.2.

To design the natural death operators, the structure of the multi-stranded DNA complexes only consists of the binding-operator since there is no output strand is expected. Other additional DNA gates are called buffering operators, which are useful to slow down the reaction rate at some particular situation, for example in an oscillating behavior such as in Lotka-Volterra model. It is important to maintain the low reaction rate so that no species dies prematurely from the system. However, we assume that all reactions in our design occur under the same environmental conditions and the reaction rates for each DNA strand is also assumed to be equal. A multi-strand DNA complex that can reversibly bind to every single strand DNA can be implemented to slow down the reaction as they are competition between the main operators and the buffering operator. The buffering reaction is reversible, so it does not permanently reduce the amount of the input strands but only delay the whole reaction. The buffering operator design is similar to the natural death with additional toehold domain on the right-end to facilitate reversible reactions.

3.4 Simulation Results

To evaluate the implemented model, we utilize a software-based simulator called Microsoft Visual DNA Strand Displacement and to compare the obtain results with the mathematical implementation by using Matlab. Visual DSD is a tool for programming language of composable DNA circuits that includes basic elements of sequence domains, toeholds and branch migration (65). It compiles a collection of DNA strands into a reaction network based on DNA strand displacement. The software also includes a deterministic and stochastic simulator to graph species population over time. It work under main assumption that there is no secondary structure of the involving strands. The usage of this simulator at this stage was motivated by previous works that has been run and verified by using this simulator, including the actual implementations of DNA strand displacement-based systems (17, 60).

There are two different scenarios taken into considerations. First, the case of partial interactions among antibodies only and without any antigen presents. Furthermore, the affinity values are integer and given fix, and there are two cases where natural death operation occurs and does not occur. Second, the case of full interactions among antibodies and between antibodies and antigens. The affinity values are real numbers given randomly, and there are two cases where natural death operation occurs and does not occur. Figure 3.5 and 3.6 show the implementation results from the first scenario. As expected, the DNA-based Immune Network model shows an oscillatory behavior since all antibodies interact with equal affinities and there is no stimulation from external by any antigen. The left-side of the figure shows the implementation of the DNA-based model, while the right-side shows the implementation of mathematical model by computer programming. The time unit and population is arbitrary, on the real experiments it may vary depending on reaction rates and concentration. The small differences between numbers of population and simulation time are neglected as at current time we are only interested in the behavior of the system. The most optimized experimental results can be obtained by setting up appropriate parameters (such as reaction rates). As shown in this figure, our DNA-based implementation behaves similarly to the mathematical model.

Figure 3.7 and 3.8 shows the implementation results from the second scenario. As expected, there is one antibody rises as the best solution depending on its affinity value to the antigens. In this case, multiple antigens are given as stimulus and there is variation on the affinity values as well, and we confirmed the result of our implementation. Similarly, we compared the result from our DNA-based implementation to the mathematical model implementation. Both systems show a similar behavior under the same initial condition.

In summary, these results demonstrate the potential of our model to be implemented in the DNA-based decision-making capable system. The system's capability to infer a different answer under different stimulus shows the adaptivity of our model. This architecture is expected to work in a dynamic environment, where the environmental conditions or given problems that may change over time. The system also accepts multiple numbers of antigens. The affinity matrix can be further optimized in-silico by properly adjusting the initial condition of input strands.

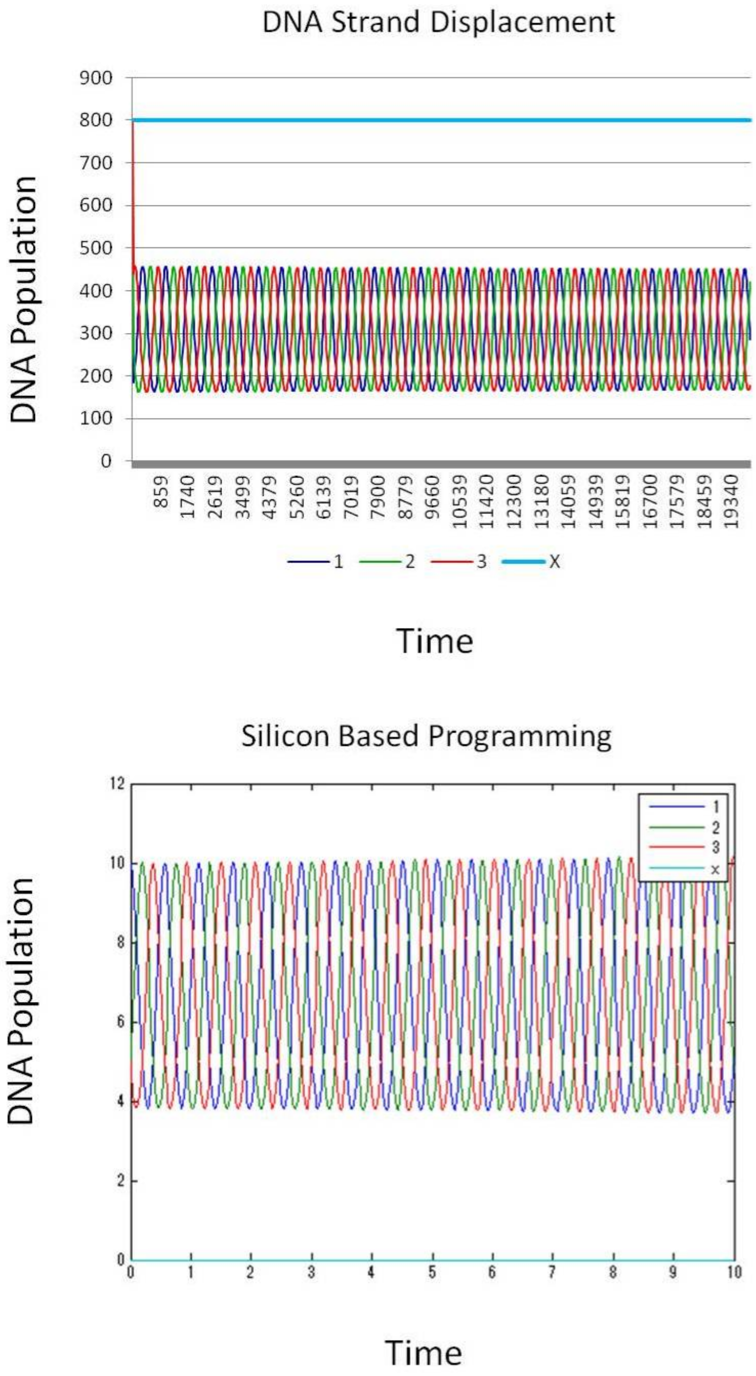


Figure 3.5: Simulation results of partial and ideal interaction among antibodies

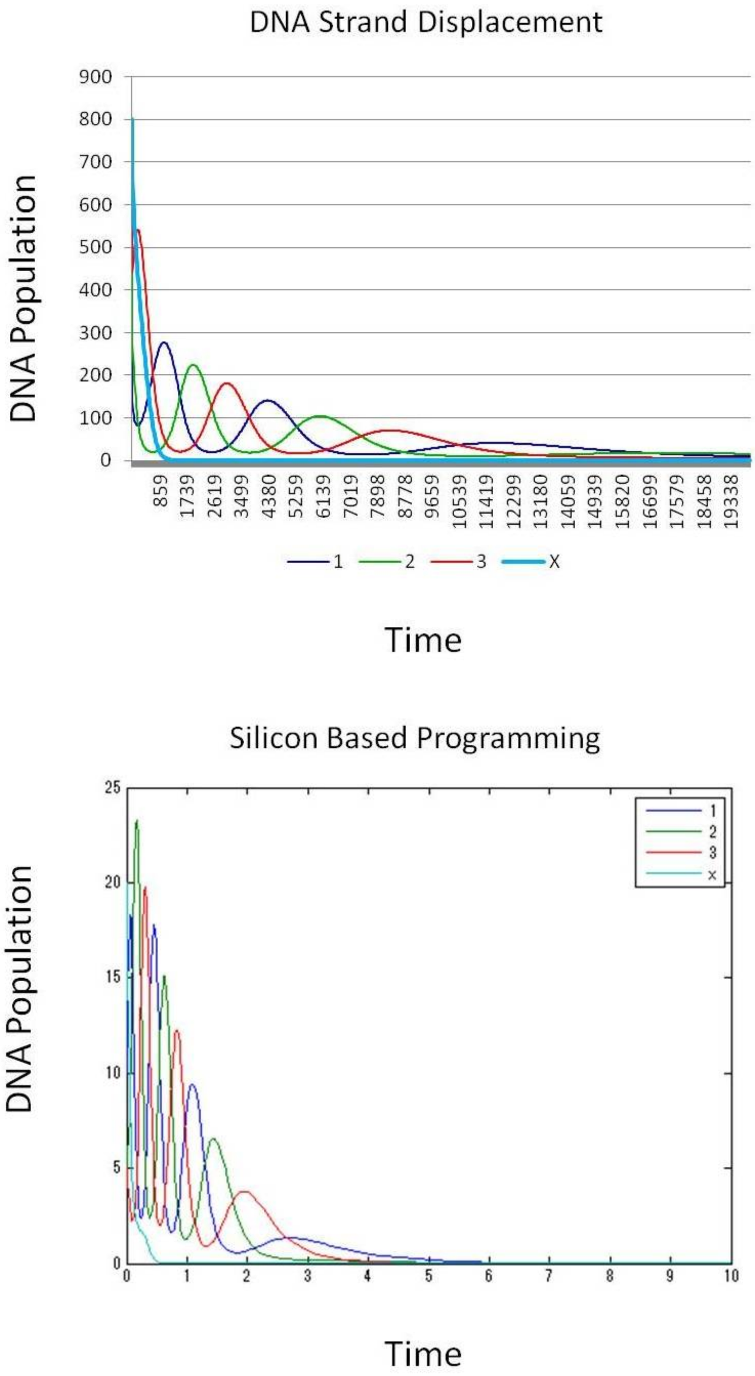


Figure 3.6: Simulation results of partial and ideal interaction among antibodies

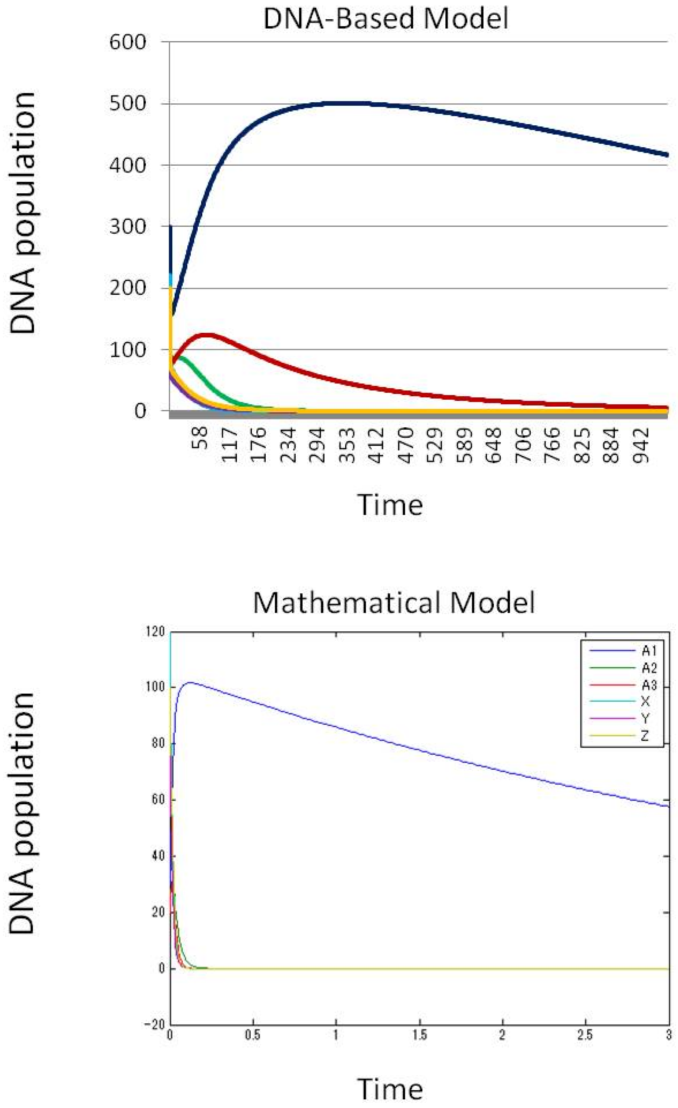


Figure 3.7: Simulation results of full interaction among antibodies and antigens

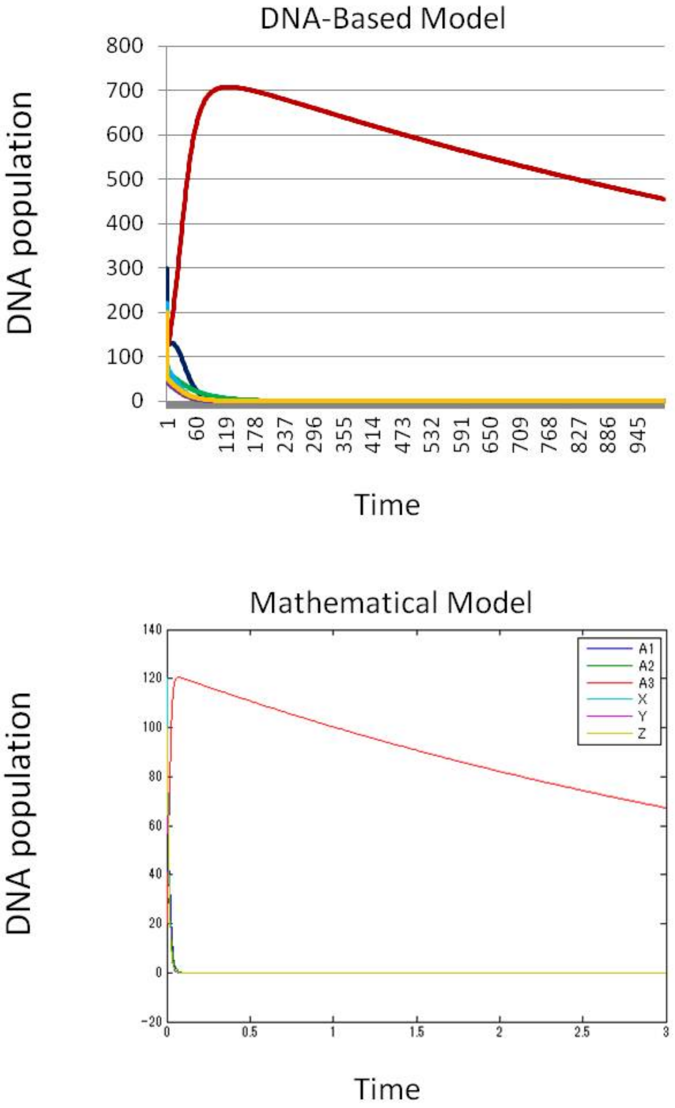


Figure 3.8: Simulation results of full interaction among antibodies and antigens

4

Ant System-Based DNA Circuit

The main challenge in developing of evolvable, autonomous, and programmable biomolecular machines is to introduce the ability to cope with external changes. In this study, we use DNA strand displacement as the main mechanism for modeling a complex-computation capable DNA circuit. Particularly, we focus on a system that can be reinforced to make intelligent decisions. The goal is to design a reactive synthetic bio-molecular system that is also adaptive to external stimulus. An instance of nature-inspired computational algorithms, namely the ant food-foraging system, has inspired the design in this work. It also incorporates the usage of DNA-based geometrical components or nanostructures, termed DNA origami. We verified the correctness of our algorithm in-silico through quantitative measurement of reaction kinetics. From the obtained results, it is indicated that the circuit can respond correspondingly regardless of the initial conditions, with some limited thresholds. This is in contrast to the currently available DNA strand displacement systems that are dependent to their initial conditions and can only be used for once. The potential applications include decision-making capable machines, and reusable DNA circuits.

4.1 Introduction

Adaptation is a key for survival. For decades, the idea of evolvable machines has intrigued many scientists, both from computer science and biology. The emerging study of molecular robotics and computer has been an active field of research in recent years. The latest progress of DNA nanotechnology has introduced more sophisticated biomolecular systems that are autonomous (20, 21), programmable (66), and both autonomous and programmable (26). The next challenge is, as discussed by Murata et. al.: Can we build artificial molecular systems that learn to adapt their environmental condition? If it is possible what kind of design fundamentals should be taken into consideration? This feature is important for designing biological machines at the molecular level, such as molecular robotics (25).

Due to the programmability and predictable behavior of the Watson-Crick pairing, DNA has raised as a substrate of choice for building various interesting applications at the molecular level (1, 2). Progress toward this direction has been made in the rational design of DNA-based components, such as sensors and motors (67, 68), circuits (13, 51, 53), and structures (7, 69). The appropriate biomolecular information processing technologies are on demand as the complexities of such applications grow. One of the most successful approach in delivering DNA-based dynamical systems is the toehold-mediated and three-way branch migration DNA strand displacement system (termed DNA strand displacement or DSD), for its modularity and robustness for computations involving biochemical materials (1).

Surprisingly, only a few have attempted to design DNA circuits by taking inspiration from how natural systems process and carry information. One of the most seminal demonstrations to date is the brainchild of Qian and Winfree (52). Having been inspired by the way human brain performs computation, they developed a DNA system that can be "trained" to memorize patterns and to deduct the correct answer when incomplete stimulus are present. Soon after, Genot et al. dramatically reduced the number of required operators by their simplified design, thanks to the cues from the distributed competition for shared resources. This mechanism can be observed in various population dynamics, for example during the enzymatic replication process (70).

Soloveichik et. al. conceptually outlined general designs of DNA strand displacement-based systems by utilizing a mathematical-chemical notation, termed Chemical Reaction Networks (CRNs) to reverse-engineer chemical reactions into DNA primitives. In this work, they demonstrated various classes of dynamical systems, such as the Lotka-Volterra model, limit cycle, and state machine, which can be implemented over DNA strand displacement semantics (10). Additionally, Mardian et al. formulated the correlation between the DNA-based implementation and the mathematical models based on a specific form of nature-inspired computation, namely the Immune Network Theory. Theoretically speaking, this approach can be employed to develop synthetic biomolecular systems that are capable of decision making, directed by the mutual interaction between the involving species (71).

In this chapter, we investigate another algorithm based on nature-inspired computation that can be utilized for designing a DNA-based circuit that can be reinforced to learn to adapt to a dynamic environment or sudden changes. We take an inspiration from the by positive and negative feedback mechanisms and distributed-cooperative interaction underlying the colony of ants. For decades, this computational metaphor has been investigated in computer science and applied mathematics. To date, there have been various formulations on its mathematical model as well (72).

Ants are simple creatures with physical limitations. For example, some species of ants are blind and have very few brain cells compared to humans. However, in a group, they are capable of performing intricate tasks with a complexity beyond their individual capacities. For instance, despite the absence of a central leader and movement coordinator, in collecting food, a colony of ants is always able to find the most effective route and the shortest-possible path for transporting food-loads to the nest. Interestingly, this process emerges from distributed interaction between ants, without any individual awareness and full comprehension of the overall happening process. An ant behaviorally acts simply to respond the local encountered stimulus.

This process can be adopted to design DNA-based interaction in a test tube. DNA strands can be treated as "self-interested agents" coexist in a swarm population with no awareness of each other presence. They are "selfishly" attracted to their complementary sequences through hybridization or toehold exchange. These massive interactions, which can be directed through appropriate design of their reaction pathways may lead to the establishment of equilibrium states. We show how the ant-foraging behavior can be employed as a basis for designing DNA strand displacement systems reaction pathways throughout this chapter. The idea is to exploit the distributed interaction between ant-inspired agents (which are represented by DNA strands), the spatio-temporal aspects of designed circuit by incorporating recent trends in both structural and dynamic DNA nanotechnology, as well as the principle of indirect communication mediated by the environment. Our model is majorly inspired by the recent architectures, termed spatially localized DNA circuits. Being different with free floating systems, DNA complexes or DNA gates are immobilized as stators on DNA origami. In principle, this may speed up the reaction and increases the DNA circuit's efficiency (73, 74, 75). On important feature of our design is that this architecture enables straightforward visualization of ant-based DNA agents traveling the structure by the molecular-walker mechanism.

To evaluate our model, we measure the reaction kinetics in-silico. The obtained results suggest the robustness of our model that makes it suitable for dynamic environments. This DNA circuit model can be reinforced to learn to make stochastic decision, and it is independent to the initial concentration to input signals under certain threshold. This can be achieved thanks to due to the adaptability to corresponding unknown environmental stimulus. The potential applications of the system presented in this study include the novel design of reusable DNA circuits.

The rest of this chapter is organized as follows. Section 2 describes the brief theory underlying nature-inspired computation models on which our DNA circuit design is based. Section 3 explains the main architecture of the DNA strand displacement systems, including stochastic-based operation. Section 4 discusses the implementation details, followed by simulation results. Finally, discussions are presented in Section 5.

4.2 Artificial Ant Systems

Nature-inspired computation, or swarm intelligence in computational science, is a field of study with interests in algorithms that mimic the way of natural systems carry information via interaction networks and directed by simple behaviors of involving individuals, or particles or agents. From this local interaction between

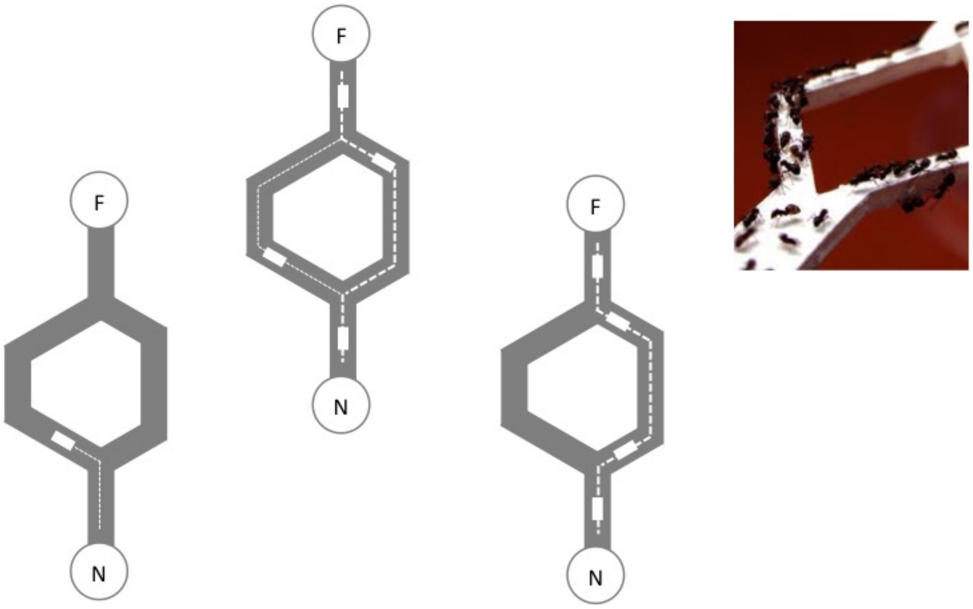


Figure 4.1: Double bridge experiment

agents, global system dynamics gives rise. In this study, we focus on a particular class of nature-inspired systems based on the ant-foraging behavior that may be utilized for designing DNA based computation.

Like many other colonies of insects, ants are social creatures that perform their tasks in highly structured coordinated manner. Ants are capable of performing complex tasks beyond their individual limit in a group, for example during the coordinated foraging and food gathering. When searching for a new food source, there is no central leader that directs the movement of the colony. However, despite this lack of coordinator, ants are still capable of finding collecting food in the most optimum way. This behavior has motivated many studies in computational science, in order to explain the means by which simple creatures such ants can exhibit that high-level organization.

The ant-foraging system's key of success relies on an indirect communication mechanism mediated by changes in the environment, which is referred to as "stigmergy". Ants communicate in a form of coordination involving chemotaxis-based process, instead of communicating via direct signal transmission or relying on visual perception. Ants, when explore their surrounding, deposit a certain amount of chemical compound, termed pheromones, which attracts other ants in the colony by its odor. An individual of ant is not aware and does not have any comprehension of the global view of the task being performed. They are instead "behaviorally-programmed" by a simple set of rules, such as moving to follow the pheromones trails, and leaving pheromones trails on area they have traveled. Thus, ants decision is biased to pheromones exposure. The movement of the ant colony is driven to the path traversed by most ants, as this path supposedly contains the greatest amount of pheromones. This can bee seen as a positive feedback mechanism of the ant systems. On the contrary, the pheromones existence is also dependent to time, as they vaporize slowly. When pheromones on a particular path are not constantly updated, or are updated with much slower rate, the trails disappear and are no longer followed by ants. This can be seen as a negative feedback of the system.

The following details explain this principle. Suppose there are two different paths of different length with pheromones trails. The path with the longer distance takes more time to update compared to one with the shorter distance. Intuitively, given a time period, the amount of pheromones on the shorter path will be accumulated faster, which reinforce ants to choose to travel along that particular route. Moreover, as the evaporation rates are the same for both paths, pheromones along the longer one disappear faster than the shorter one. As the result, the whole colony will prefer to follow the shortest path in the end.

Figure 4.1 illustrated the "double-bridge experiment" which is an early example of the ant foraging behavior study (76). A colony of ants have to collect foods at point F by traveling from point N. There are two possible branches of different lengths along the route. The ants have to choose the shortest path, which is initially unknown to them, in order to maximize the amount of transported food in a given time. Ants produce pheromones trail to mark their chosen path as they traverse the bridge (left-side). The first ant chooses an option at random when it encounters the branch. It could possibly the longest one since there is no pheromone trail for it to follow. On the contrary, the next ant's choice is affected by the marked by pheromones trail of the previous ant. In early time, since there are only a few of ants have chosen the route, the secretion of pheromones along both routes are relatively low. Therefore, there is still a high probability that the next ant would choose a path differently (middle). Eventually, the accumulation of pheromones trails along the short route will be a lot faster than in the longest one. On the other hand, pheromones trails in the longest path evaporate quickly. As this keeps continuing, ants will be reinforced to follow the shortest path (right-side). Inset: real ants during the food searching as depicted by the double bridge experiment. Image was taken from http://archive.cosmosmagazine.com/news/simple-rules-smooth-traffic-ant-highways/.

$$p(r,s) = \frac{[\tau(r,s)] \times [\eta(r,s)]^{\beta}}{\sum_{u \in M_k} [\tau(r,u)] \times [\eta(r,u)]^{\beta}}, s \in M_k \tag{4.1}$$

$$\tau(r,s) = (1-\rho) \times \tau(r,s) + \sum_{i \in P(r,s)} \tau_o, 0 \leq \rho \leq 1 \tag{4.2}$$

This self-organizing principle has been studied formally in computational science and applied mathematics. They have been successfully implemented in various applications such as machine learning and robotics. Equations 4.1 and 4.2 show the mathematical formulation of the ant systems (76) (72). The probability of an ant choosing the path from r to s is shown by Equation 4.1, where $\tau(r,s)$ is the amount of pheromones on the corresponding path, $\eta(r,s)$ is the desirable weight function (i.e. inverse distance as heuristics), β is the importance parameter against the pheromones, and M_k is ants' working memory, which defines the paths that have not been visited. The update in pheromone concentration after visiting is shown by Equation 4.2, where ρ captures the evaporation rate and τ_o represents the amount of pheromones mark left by an ant at one time unit. Therefore, the update at one path is equal to the summation of all ants i choosing to travel through that path (r,s).

4.3 DNA-Based Architecture

4.3.1 DNA Strand Displacement Operator

Any DNA strand displacement system can be represented by using the mathematical-chemical notation referred to as Chemical Reaction Networks (CRNs) (10). The

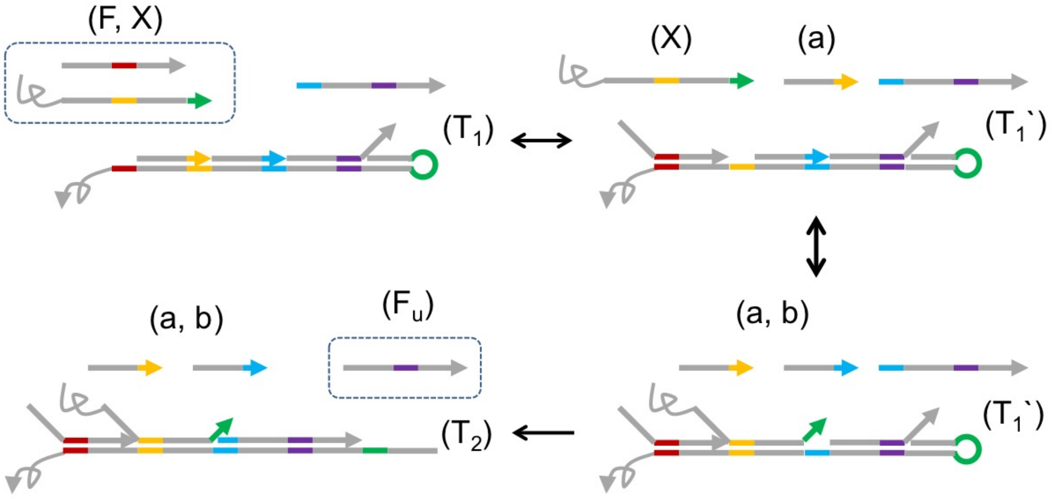


Figure 4.2: Signal transformation by DNA strand displacement reaction

generic implementation of the operator in this work is as represent by a chemical reaction equation $\mathrm{X_1 + ... + X_m + T_1 \xrightarrow{k_1} Y_1 + ... + Y_n}$. It may be translated as a programming instruction that of an equivalent transformation of m-number of input signals into n-number of output signals by a gate T_1 with reaction rate k_1. From this abstract level, there are various design strategies can be employed to obtain reaction controllers, or referred to as gate motifs. The main principle is to encode input and output signals into single-strand DNA and gate motifs into multi-strand DNA complexes that are pre-hybridized before the intended reaction.

In this study, our objective is to outline the conceptual design of DNA circuits rather than to demonstrate detailed the wet-lab implementation. A simple possible coding of DNA strand displacement primitives that is easy to understand is chosen, without taking into excessive considerations of its actual limitations (e.g. possible of crosstalk, leakage reaction, and so on). With a slight modification from previously described structures (64), we define out DNA strand displacement semantics. The reaction kinetics of this particular design has been studied previously (10). It is important to mention that this implementation may not be unique, one can also implement our design by using different structures. For example, the above-mentioned chemical reaction equation may be implemented

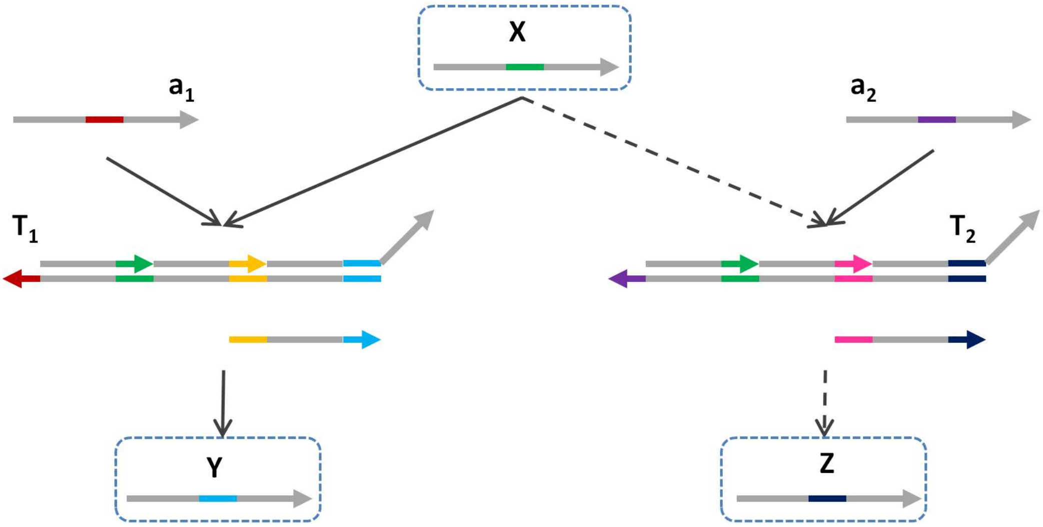


Figure 4.3: Stochastic DNA operation

by using gate motifs with similar behavior from (13, 53, 77).

Figure 4.2 shows an example of an arbitrary signaling reaction. Suppose there are two input signals (F, X) and a gate motif T_1. By design, gate T_1 consists of one multi-strand complex and one long single strand as the trigger (top-left). Presenting the gate in the system means adding both strands into the solution. As signal F and gate T_1 share a free toehold, the reaction begins and they reversibly reacts with each other. This reaction is followed by the displacement of auxiliary strand a, which modifies of the gate structure into intermediate gstate T_1' (top-right). Then, signal X is now able to interact with the gate, as a new free toehold is exposed, and a similar process occur to displace auxiliary strand b (down-right). As the final step, the trigger irreversibly binds, releasing the output signal F_u, and opening the loop structure by the end of the gate. The gate is converted into a different gate T_2 and reveals a new toehold. Signal X remains active since it still consists of one free toehold (it is now referred to as X'). In summary, the whole reaction may be seen as transformation of species F, X, T_1 into F_u, X', T_2. This will be the basis or the main operation of reaction controllers for the DNA-based agent walking mechanism that implements our ant system-based DNA circuit.

4.3.2 Stochastic DNA Operator

The stochastic operation can be implemented by adjusting the amount of a particular gate motif, when there are at least two transformation rules are involved with the same input signal. This will increase the likelihood of firing the corresponding output signal. The following example illustrate this mechanism: suppose there are two transformation rules with equal rates: $\mathrm{X} + \mathrm{T}_1 \xrightarrow{k} \mathrm{Y}$ and $\mathrm{X} + \mathrm{T}_2 \xrightarrow{k} \mathrm{Z}$, controlled by gate motifs T_1 and T_2 respectively. Assuming the amount of T_1 is equal to T_2, the probability of both reactions occurring is also uniform by 50:50 ratio. However, if the amount of T_1 changes, for example, to three times that of T_2, the probability of the occuring reaction changes to approximately ratio 75-25 ratio.

In many implementations of DNA strand displacement systems to date, the dynamic adjustment of the amount of gate motifs requires external assistance from human operator. On the other hand, manipulating single-stranded DNA is easier to than multi-stranded complexes. One possible way to do so suggests a technique termed the buffering operator, which makes a particular gate motif "inactive" until corresponding releasing signals are present. This simulate a design of an operation that produces a gate motif instead of a signal. However, cascading this mechanism into multi-layered reactions is still a difficult task (64). In this study we suggests the implementation by using two AND gates (or join operation), as shown in Figure 4.3. We call this bio-molecular procedure as "Stochastic DNA operator". Our implementation suggests the application in any DNA-based stochastic selection and decision-making. A stochastic transformation of an input signal X into two output signals Y and Z may be achieved by combining two gate motifs that implements instructions $\mathrm{a}_1 + \mathrm{X} + \mathrm{T}_1 \xrightarrow{k} \mathrm{Y}$ and $\mathrm{a}_2 + \mathrm{X} + \mathrm{T}_2 \xrightarrow{k} \mathrm{Z}$. There are two new signals a_1 and a_2, referred to as implicit signals, which facilitate indirect control of the reaction kinetics. We assume that both gate motifs have the same displacement rate. So, given the initially equal amount of gate motifs $T_1 = T_2$ and $T_1, T_2 >> a_1, a_2, X$, intuitively we can infer that the implicit signal a_1 and a_2 control the amount of gate motifs available for input signal X to react during the second step of the reaction. Therefore, the probability of both transformations are roughly determined by the ratio between the implicit

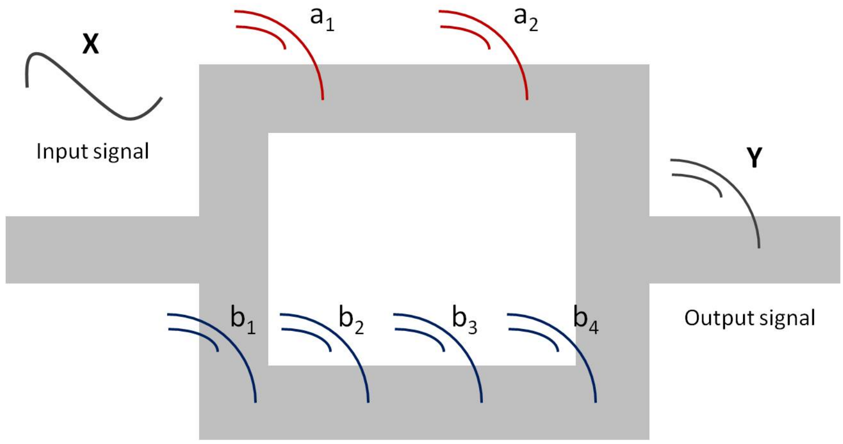


Figure 4.4: Design of the double bridge-inspired DNA circuit (top-view)

signals a_1 and a_2. The direction of the reaction can be determined by adjusting the amount of signals a_1 and a_2. For example, the reaction between X to Y is more probable to happen (solid line) than the reaction between X to Z (dashed line) by stimulating a larger amount of a_1 than a_2.

4.3.3 Localized DNA Architecture

DNA origami is a nanostructure composed of DNA folded together to assemble into certain shapes of bio-molecular landscape (7). Over the years, it has been utilized in various applications in nanotechnology and also to enhance DNA-based computation (26). DNA strand displacement reactions can be sped up via localized hybridization by tethering DNA strands on the origami surface, and by employing molecular walkers traversing the DNA stators (73). We utilized the similar architecture for simulating of our design because of the visual resemblance between the DNA walkers-origami circuit and ant colony system, where artificial ants are equal to DNA walkers and DNA origami is as the working environment).

A DNA structure design that is inspired by the ant system and resembles the double-bridge experiment is shown in Figure 4.4. Gate a_1 and a_2 represent a set of gate motifs designated as stators of the first branch; whereas gate b_1, b_2, b_3, and b_4 represent a set of gate motifs designated as stators of the second branch. The number of the gates on each branch are the density value or weight function of those branches. They can be assigned with arbitrary integer numbers. We have chosen the minimum amount for simplification reason. Both branches accept signal X as their input and transform it into output signal Y through several intermediate reactions (the more stators, the more reactions are involved). This simulates the walking of the artificial ants. Furthermore, this double-bridge structure can be treated as a molecular building block for construction of arbitrary two-dimensional landscapes consisting of paths with different length. The objective is that, without any prior information given, the artificial ants are supposed to find the most optimum path that allows the collection of better weight.

DNA origami can be built by folding a long scaffold strand with short staple strands that can hold the scaffold in a particular arrangement. There have been some software developed to generate the most optimum sequences of each staple, and they can be annealed together through a one-pot reaction to achieve the designed structure. In case of localized DNA circuits, arbitrary staple strands along the branches may be extended as DNA gates that act as stators for computation.

Thus, we can assign different gate densities to each branch, which also referred to as the weight function. Each branch may take the same input signal but produce different output signals. The number of intermediate reactions along the branches may vary, and it represents the goodness of the branch. The computation is restricted spatially since DNA gates are immobilized on the origami. To avoid the cross-talk the distance between gates or stators can be adjusted accordingly, so that it is long enough for the walkers to not jump directly to the end point, and at the same time it is short enough to cascade the reactions. The following section discusses further implementation details of the ant system-based DNA circuit.

4.4 Implementation and Results

4.4.1 DNA-Based Implementation

In order to translate the mechanism of the ant foraging behavior system into our DNA circuit design, first we need to decide the representation of required components. The stigmergy, or the indirect communication mechanism of ants, relies on the environmental changes. In this case, it involves spatial localization. It means thaat ants need to walk from their nest to the source of food, and the amount of pheromones trails depends on the distance between those two points. Intuitively, this can be represented by a landscape of DNA origami consisting of two arbitrary points denoted by the nest (starting point) and the source of food (finishing point). DNA strands that traverse the origami as molecular walkers represent the artificial ants. Analogous to the process where ants secrete pheromones, these DNA signals also produce other output strands as they walk through the stators.

Consequently, this procedure can be captured by a reaction that may be defined as the consumption of previous pheromone signal, ant signal, and stator, resulting in the production of a new pheromone signal, catalysis of ant signal, and another fuel signal to drive the movement of the walker. In general, Figure 4.2 shows gate motifs that can control these operations. This implementation can replace the stators abstraction as shown in Figure 4.4.

$$\begin{aligned} F_a + X + a_1 &\rightarrow F_a + X^* + f_{a2} \\ f_{a2} + X^* + a_2 &\rightarrow F_a + X^{**} + f_{a3} \\ f_{a3} + X^{**} + y &\rightarrow F_a + X^{***} + Y \end{aligned} \tag{4.3}$$

$$\begin{aligned} F_b + X + b_1 &\rightarrow F_b + X' + f_{b2} \\ f_{b2} + X' + b_2 &\rightarrow F_b + X'' + f_{b3} \\ f_{b3} + X'' + b_3 &\rightarrow F_b + X''' + f_{b4} \\ f_{b4} + X''' + b_4 &\rightarrow F_b + X'''' + f_{b5} \\ f_{b5} + X'''' + y &\rightarrow F_b + X''''' + Y \end{aligned} \tag{4.4}$$

$$\begin{aligned} F_a &\rightarrow \emptyset \\ F_b &\rightarrow \emptyset \end{aligned} \tag{4.5}$$

Equations 4.3 to 4.5 describe the reactions involved in our DNA circuit design. Equation 4.3 and Equation 4.4 capture the ants walking mechanism and positive feedback on both branches respectively. Equation 4.5 depicts the negative feedback or the evaporation of pheromones. In this example, the second branch consists of a larger number of gates so it is considered to be better as it has a better weight than the first one. Note that it is possible to assign arbitrary values by design. However, the main purpose of this study is to demonstrate the capability of our DNA circuit to make decisions with a better weight and to eliminate the less feasible choices. A larger landscape can be constructed from this structure. One unit of the bridge can be treated as a building block, as shown in Figure 4.4.

The pheromone signals for selection of each branch are represented by F_a, F_b. In the stochastic DNA operator, as depicted by Figure 4.3, the implicit strands control the direction between two possible reactions. The artificial ant is represented by signal X, with additional mark ' and * to represent intermediate states when the agent is in the middle of the walking mechanism as it bound to particular stators. It will be released back into the solution, when it arrives at the destination point. Signal $a_1, a_2, b_1, b_2, b_3, b_4, y$ represent the stator or DNA gates along branches. Signal $f_{a2}, f_{a3}, f_{b2}, f_{b3}, f_{b4}, f_{b5}$ represent non-stochastic pheromone signals that act as fuels to drive the ant agent's movement from one stator to another through the chosen path. Signals for the downstream reactions are produced as the output from upstream reactions to ensure the executions are performed in order. Otherwise, they can be provided as the input with a unique design for every gate.

The main difference between the natural ant system and our DNA circuit lie in the heuristics use the feedback parameter. The original system suggest the utilization of time and distance as heuristics, while our model uses the amount of gates or stators. Consequently, the better weight of one corresponding path means the more amount of gate motifs assigned to that path. One may assume that this principle is trivial, as it is easy to think that a larger number of gates will produce more signals. However, the result will vary from time to time since biomolecular reactions do not allow decision making or exclusive selection, especially when there are only few differences between gates numbers. On the contrary, on the basis of small differences between branches, we demonstrate that our design can distinguish the best path from the less feasible ones. Our design promotes

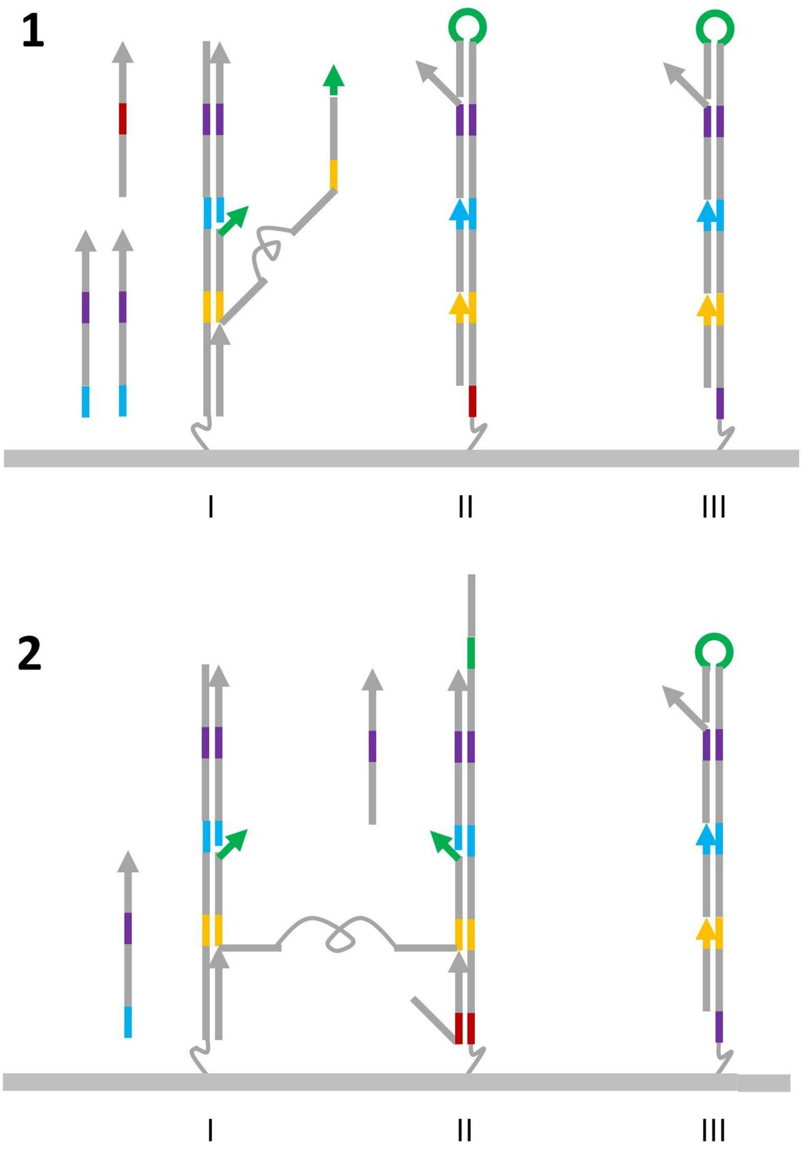


Figure 4.5: Design of walking mechanisms (step 1-2)

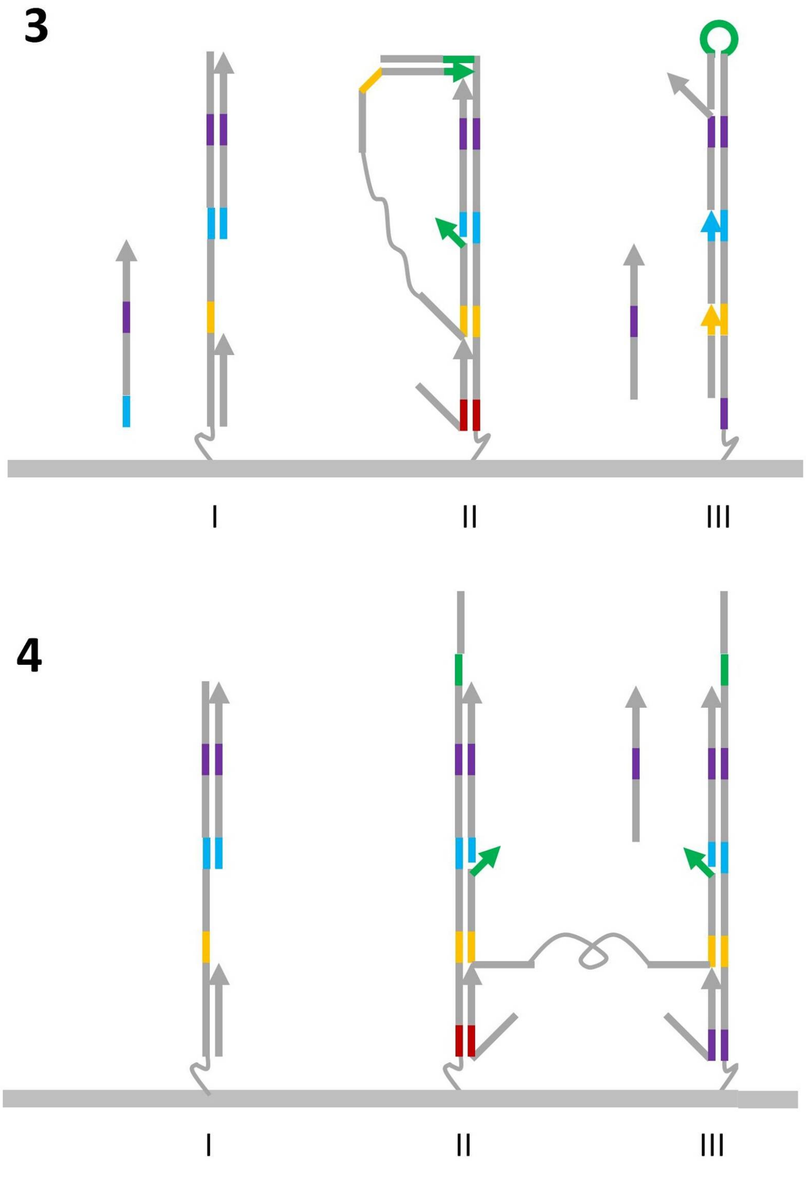


Figure 4.6: Design of walking mechanisms (step 3-4)

improved selection of the best option and elimination of the remaining options, by using a metaphor learned from the positive-negative feedback mechanism of pheromone-based communication. In this chapter, a two-choices ant-bridge systems is presented. In principle, this model can be extended into multiple choices systems without further significant revision on its design.

The mechanics or the walking mechanisms can be implemented in various ways, for example, by using hairpin fuels (9) or DNAzyme (68). Our design has a special need as it requires different signals to mediate the indirect communication. Therefore, we propose an alternative or extended walking mechanisms as depicted by Figure 4.5 and Figure 4.6.

As shown in Figure 4.2, gate motifs are tethered as stators to DNA origami (side-view). Suppose there are three different types, with stator I being the initial point, stator II being the stochastic stator and it represents the branch for selection in the double-bridge system, and stator III being the non-stochastic stator and located in the remaining parts of either path. The stochastic stator II determines the chosen path of the DNA walker. The non-stochastic stator III is unique to each path. Figure 4.2 shows the details of the reactions. The first input is either the fuel strand from the previous stator's output or the pheromone signal for the stochastic stator. The second input is the DNA agent with a flexible region connecting two equivalent legs. The flipping when the walker moves forward is possible because there are similar regions also present in the staple strands extension to DNA gates.

4.4.2 Correlation with Mathematical Models

The kinetics of pheromones signal at a branch r is defined by the following equation: (The first and second terms are derived from Equations 4.3 and 4.4, while the third is from Equation 4.5.

$$\frac{d(F_r(t))}{dt} = -(q\times F_r(t)\times X(t)\times r_1(t))+\sum_{j=1}^{N}(q\times f_{rj}(t)\times X^*(t)\times r_j(t))-(q\times k\times F_r(t)) \tag{4.6}$$

Assuming that rate q is uniform for the whole strand displacement reaction, and stators r_j (hence the open-loop f_{rj}) are available in large amounts $n \to \infty$, and signal X is not consumed by the reaction, their values can be replaced by the constant δ. Therefore, Equation 4.6 above can be simplified into Equation 4.8. Intuitively, this denotes the parameter of pheromone update, where evaporation rate $k<$ constant δ to satisfy Equation 4.2.

$$\begin{aligned}\frac{d(F_r(t))}{dt} = -(k \times F_r(t)) + (X(t) \times \sum_{j=2}^{N}(f_{rj}(t) \times r_j(t))) \qquad (4.7)\\ \frac{d(F_r(t))}{dt} = -(k \times F_r(t)) + \delta\end{aligned}$$

For the probability equation, the stochastic selection is based on the first-line reactions of Equations 4.3 and 4.4. Their implementation satisfies the stochastic DNA operation as in Figure 4.3, with extension to multiple output signals. The kinetics of the sum of output signals P_r at one branch, and total output signals P for whole branches is denoted by following equation.

$$\begin{aligned}\frac{d(P_r(t))}{dt} &= q \times F_r(t) \times X(t) \times r_1(t) = \alpha \times F_r(t) \qquad (4.8)\\ \frac{d(P(t))}{dt} &= \sum_{u=1}^{N} q \times F_u(t) \times X(t) \times u_1(t) = \sum_{u=1}^{N} \alpha \times F_u(t)\end{aligned}$$

According to this equation, the probability of selection of branch-r at a given time t can be calculated as Equation 4.9 below. It is clear that the coding correctly implements the original mathematical model as in Equation 4.1, where the constant δ incorporates the desirable function and importance parameter that are the same for all available selections.

$$p(r,t) = \frac{\alpha \times F_r(t)}{\sum_{u=1}^{N} \alpha \times F_u(t)} \qquad (4.9)$$

Figure 4.7 summarizes the correlation between each mathematical parameter in our DNA circuit design. This graph shows the mechanism of the communication-mediated-by-environment, where the time-evolved competition between positive feedback $\delta(r)$ and negative feedback k leads to the selection of a solution with better weight function. The internal model represents the local view of each agent, without any understanding of global environmental conditions.

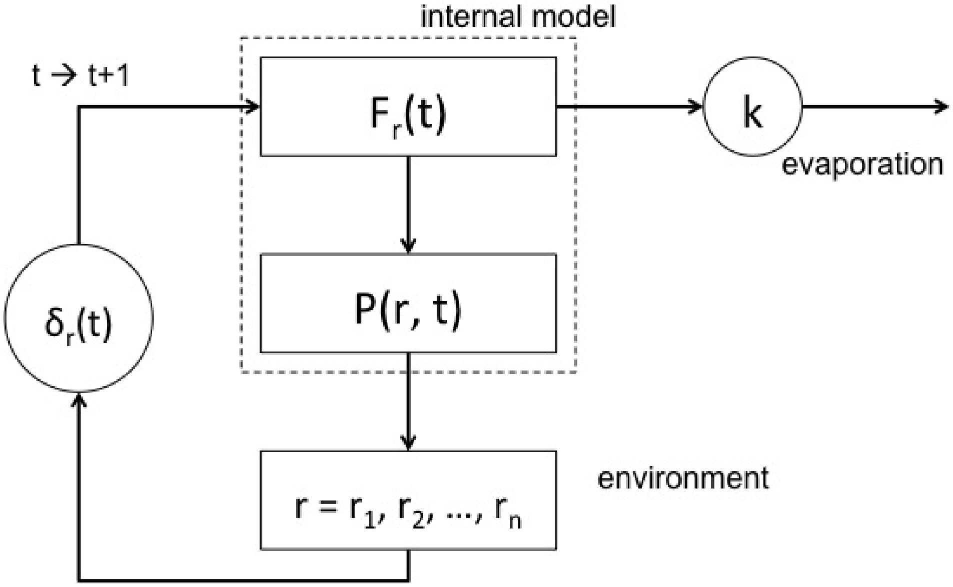


Figure 4.7: Graphical representation of the mathematical model of the double-bridge inspired DNA system

4.4.3 Simulation Results

We verified our implementation in-silico by using Visual DSD (65). This tool is a program for composable DNA circuits, that includes basic elements of sequence domains, toeholds and branch migration. Visual DSD compiles a collection of DNA strands into a reaction network based on the DNA strand displacement mechanism, and includes stochastic and deterministic simulators to display reaction kinetics. It works on the assumption that the strands do not possess any secondary structure.

This tool has been applied for running and verifying actual implementations of DNA systems (52, 53, 70), motivated the usage of this simulator in our work. Straightforward programming was obtained from chemical reaction equations as in Equation 4.3 to 4.5.

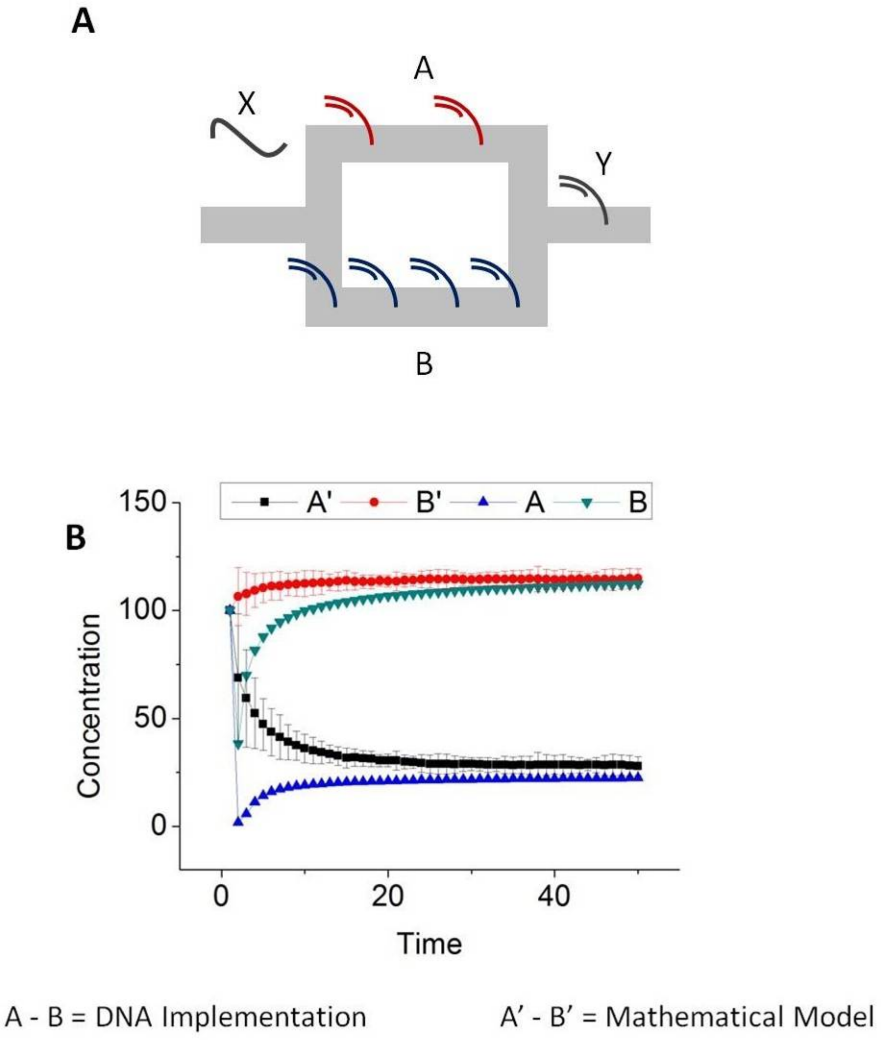


Figure 4.8: A) The double bridge structure consists of two paths selection with arbitrary weight function. B) Simulation result in for the same initial concentrations of the pheromone signal

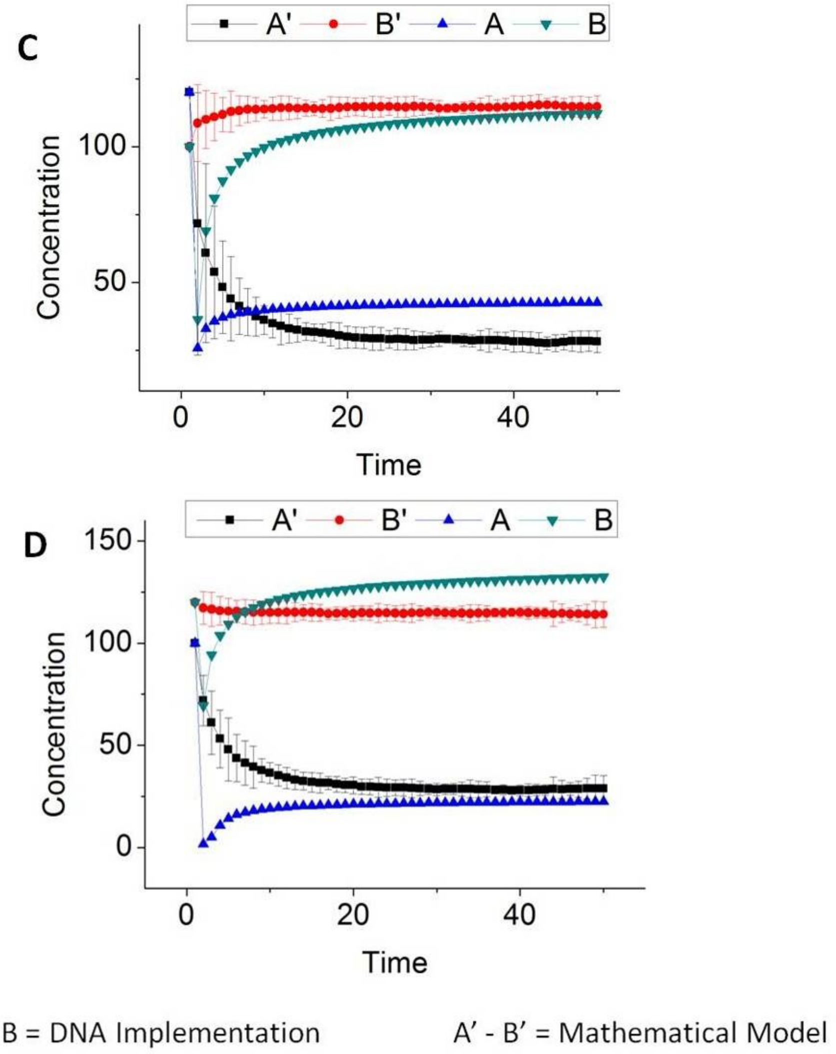


Figure 4.9: C) Simulation result for different initial concentrations of the pheromone signals, where the best selection initially has a higher concentration. D) Simulation result for different initial concentrations of pheromone signals, where the best selection initially has a lower concentration

Figure 4.8 and Figure 4.9 show the simulation results of the present nature-inspired DNA circuits design, in comparison with mathematical models. The adaptability of our design, i.e. its ability to cope with changes in the environment is demonstrated by its capability of identifying the path with the better weight function, regardless of the initial concentration. Moreover, the present DNA circuit design can approximate the implementation based upon existing mathematical models. Simulation parameters are tuned-up by iterated trials, and the best values that yield corresponding results between both models are chosen. DNA concentration is expressed in arbitrary unit. The time parameter is selected to be long enough to allow the reaction to reach equilibrium (approximately 15 hours in real experimental conditions (52)

Our design demonstrates the capability of the DNA circuit to learn to make decisions to adapt external stimuli from the environment. Given the arbitrary landscape configuration the design can infer the optimum path to reach the target. We also tested our design under different initial conditions: 1) both output signals were initially at the same concentration, 2) the expected signal concentration was initially lower than the other signal, and 3) the expected signal concentration was initially greater than the other signal. Under all these conditions, our circuit was consistently able to infer the best solution. The second and the third conditions, particularly, demonstrated the capability of our design to work correctly regardless the initial concentration of output signals. This property, which is lacking in most currently available DNA-based circuits these days, should additionally enable the potential application of our ant systems-based model for reusable DNA circuits.

Figure 4.10 and Figure 4.11 show further analysis of our simulation results. On the vertical axis (outputs) are shown in arbitrary units, either in absolute values or in normalized percentages. On the horizontal axis (parameters) are shown as ratio of concentration or ratio percentages, e.g. 1x=100nM. This value may vary depending on the experimental requirements. We tested each parameter separately and observed its effects on the output signals concentration. A higher amount of initial gate concentration and a lower amount of initial evaporation gate concentration was found to clearly produce a higher output concentration for both signals (4.10.a and 4.10b). Despite the increase in the production of

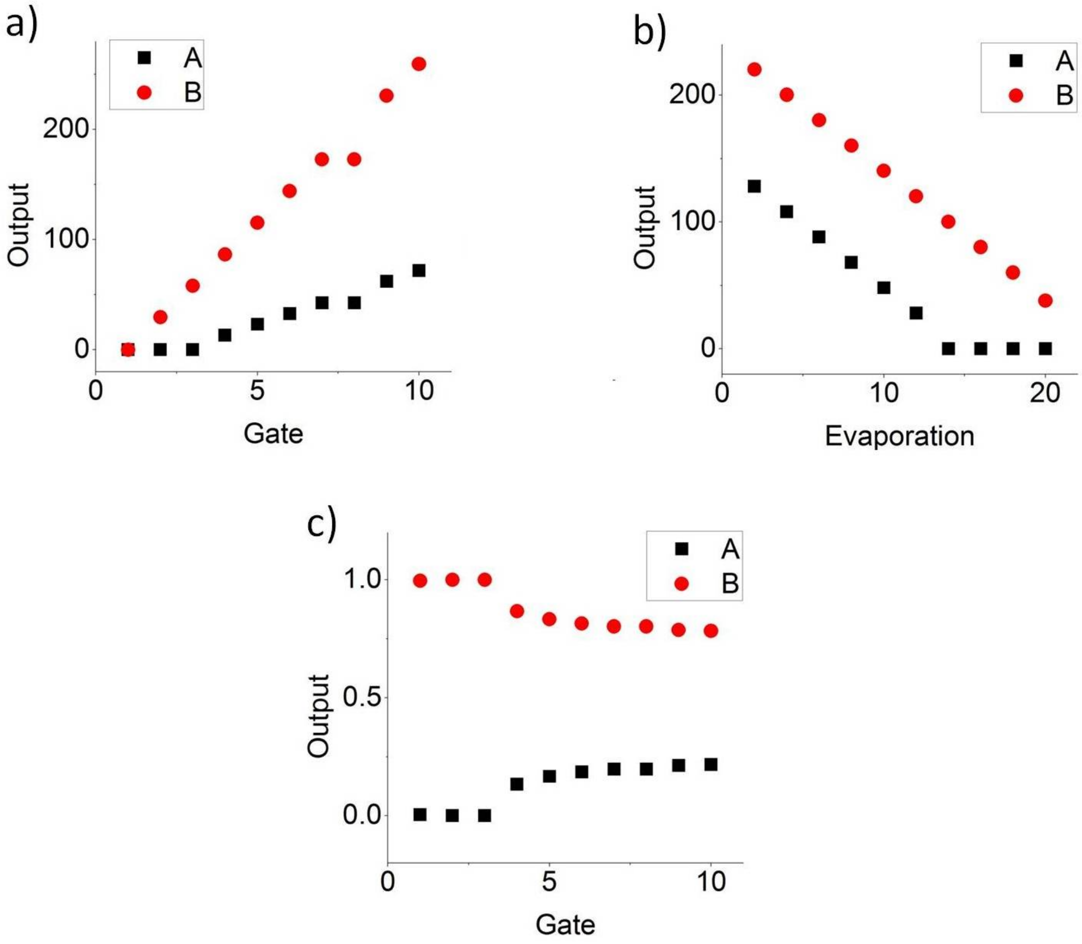


Figure 4.10: Analysis of simulation parameters. A) Initial gate concentration versus absolute values of output concentration. B) Evaporation rate versus output concentration. C) Initial gate concentration versus effective percentage of output concentration

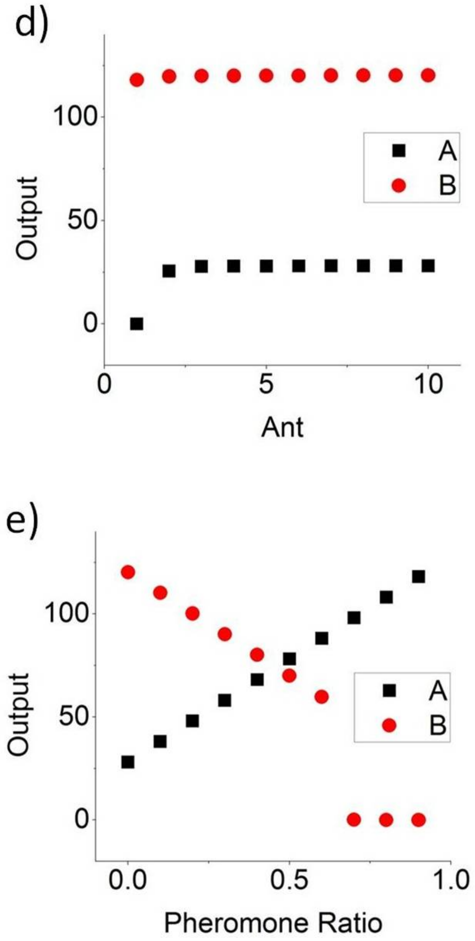


Figure 4.11: Analysis of simulation parameters. D) Number of ants versus output concentration. E) The difference between pheromone ratios versus output concentration

the less favorable signal, the percentage of the final ratio between the signals was unaffected (4.10.c). The lower initial gates, however, effectively eliminated the worse selection. However, this may result in the failure of the second iteration in reusable DNA circuits, as one possible signal completely disappears from solution. The output is not significantly affected by the number of ants (4.11.d)), as our design employs a deterministic, rather than stochastic, model for reaction kinetics. However, this is important as the catalyst for the reaction. Our design can work independently of the initial output signal concentration; however, a certain threshold value of ratio difference is required for this. For example, our circuit works effectively when the signal concentrations are the same (ratio difference = 0%). As the ratio difference percentage increases, the effectiveness of our circuit decreases. Thus, the ratio difference between the initial output signals should ideally be under 50% (4.11.e).

5

Agent-Based DNA Circuit

Coordination is one important feature in delivering distributed systems. A typical approach in developing silicon-based agents, such as mechanical robotics, is by designing individual-level behavior that emerges into one global functionality. Meanwhile, as DNA-based agents are based on chemical reactions, programming of every individual behavior still remains a big challenge. Once all reactants have been mixed into a solution the reactions occur immediately. This introduces nontrivial challenges in logical control. In this chapter, A strategy for coordinated and event-driven DNA-based systems by using a Petri Nets model is reported. First, we introduce computational primitives based on DNA strand displacement reactions. Second, we abstract their molecular implementation by Petri Nets for higher-level design. Third, we propose a model of interacting multi-agent systems based on DNA-only reactions as the main contribution of this work. The design is verified via $in-silico$ simulation. Furthermore, we show the results from initial experiments of Petri Nets operators. From the obtained results, we believe that our design strategy is suitable for coordinating interaction of distributed DNA-based systems.

5.1 Introduction

Recent progress of DNA-based applications has arrived at a level of sophistication that one can envision complex applications, such as molecular robotics and machines (2, 25, 68). For example, various designs of molecular walkers capable of traversing DNA landscapes have been proposed in (18, 20, 66), simulating the function of locomotion of mechanical machinery. A design of reconfigurable container for transporting molecular cargo, embedded with sensor and activation functions has been introduced in (67). Another demonstration of a programmable DNA-based machine that is capable of picking up, manipulating, and delivering molecular loads is reported in (24). The programmability of Watson-Crick complementary rules has made it is possible to develop bio-molecular information processing systems to control such intricate applications (1, 10, 53).

In general, there are two main features that distinguish molecular machines from their silicon-based counterparts. First, instead of hardwired within the machines, the program is incorporated through the interaction with environments (9, 50). Second, the system consists of multiple and independent particles running together in parallel, from which the summation of all individual reactions gives arise as the system global behavior (78). There have been many works focusing on the first category. However the latter mentioned is still not appropriately addressed in many studies to date, despite the fact the concurrency power available at the molecular level (54, 55).

This kind of system's efficiency and complexity can be increased by putting together modular units being responsible in different tasks together. In distributed robotics field, many researches have successfully demonstrated this design principle (79, 80, 81). However, it still remains challenge to re-engineer the same fundamental in bio-molecular systems. Programming every individual behavior is not a trivial task since it is based on chemical reactions. These reactions occur immediately once all reactants have been mixed into a solution.

In order to solve this challenge, we propose a novel design of distributed DNA-based systems, or to be specific a control architecture for DNA agents interaction. In this study, an agent is a technical term used to describe an abstract model of a molecular robot or machine based on DNA. We argue that it can be met through discretization of bio-molecular reactions by state-evolution. This is in contrast to any biochemical systems which are naturally continuous and time-evolved (82). To simulate our purpose, we employ a simple task between two DNA agents working in coordinated manner. The high-level control and coordination of molecular agents are achieved by employing agent-based model.

Along with concurrency and parallelism, agent-based models have intensively been studied in many fields including computer science, robotics, and artificial life (83, 84, 85), and also the study of coordination in biological-based and other distributed systems (86, 87). Agent-based model is said to be a superior framework, because it has several advantages over conventional system. It is suitable for building complex systems from collection of primitive components (88, 89). Its architecture can be assembled from simple and modular units, and is regulated by a simple set of behavioral rules. These rules when they are combined together will result in an intricate global function. in addition to the homogenous system, it also opens the possibility of implementing a heterogeneous one.

To model the high-level abstraction, we bridge the low-level bio-molecular operation through a mathematical modeling notation referred to as Petri Nets. This abstract model has discrete and concurrent event-driven semantics and it has been used to describe many autonomous systems behaviors, such as task planning for single-robot or multi-robot systems (90, 91, 92). It is preferable because it has easy-to-understand graphical notations, and also for its capability to represent concurrences and synchronization in a tangible formal definition. Many elegant designs can be achieved from this abstract model, such as decision making and task scheduling applications. Additionally, Petri Nets was originally designed to describe chemical reaction processes (93, 94). Therefore, it is highly suitable to model nucleic acid-based distributed systems or other systems that operate in biochemical environments.

The present work in this chapter is also motivated by recent theoretical findings on the general formalism of DNA-based dynamical networks (10, 64). In principle, arbitrary multiset rewriting systems can be implemented by one instance of prominent bio-molecular mechanisms termed DNA strand displacement (95). Consequently, this includes the formalism of DNA-based Petri Nets. To the best of our knowledge, the work presented in this study is the first to bridge the gap of such theoretical models to practical applications. Enabling a distributed control, and interaction and coordination model for molecular agents will increase the complexity of the future DNA-based systems can handle.

5.2 Petri Nets Model

We specifically work on a particular class of Petri Nets termed the Simple Marked Petri Nets. Throughout this paper, we refer to this class as Petri Nets for simplification reason. There are many other classes of Petri Nets such as Stochastic Petri Nets; therefore it should not be confused with them. Its basic transitioning rules are sufficient to model event-based systems without unnecessary confusion of more complicated rules.

Formally defined, (Simple Marked) Petri Nets is a 4-tuple of $< S, T, W, M_k >$, where: $S = \{s_1, s_2, ..., s_i\}$ is a finite set of *places*. $T = \{t_1, t_2, ..., t_j\}$ is a finite set of *transitions*. $W \subseteq (S \times T) \cup (T \times S) \rightarrow \mathcal{N}$ is multiset of arcs connecting places with transition and transition with places. Place s_i is an *input places* if there is any transition such that $t_j = \{s_i \in S | W(s_i, t_j) > 0\}$; and/or an *output place* if there is any transition such that $t_j = \{s_i \in S | W(t_j, s_i) > 0\}$. Then, s_i and t_j are said to be *connected*. $M_k = \{s_1 k, s_2 k, ..., s_{ik}\}$ is a marking state represents the current state of the system such that $M_k \subseteq S$ is any marked place of s_i that contains a token at time k. M_0 is the initial marking condition.

Petri Nets can also be represented in graphical symbols. By combining all of these components, a model of any dynamical systems can be obtained. Figure 5.1 shows the visual representation and the distributed system scenario modeled in this chapter.

the changing of the marking state through time captures the dynamics of the system. A transition t_j is fired whenever every input place s_i connected to it contains a token, such that $s_i \in M_k$. Once the transition is fired, tokens from the input places are removed and added to output places, which then changes the system state.

Figure 5.1 shows a system of two agents of coordinated manner, A and B, performing arbitrary functions. The interaction between the agents is captured by token at places s_2 and s_5. All other places represent the internal states of the respective agents. At time $k = 0$, the initial marking is $M_0 = \{s_1, s_2, s_6\}$. Because all input places connected to t_1 contain tokens, t_1 is fired, consuming tokens from s_1 and s_2 into s_3. However, t_4 is not fired since there is an input place s_5 that is not marked with a token. At the time $k = 1$, the marking state

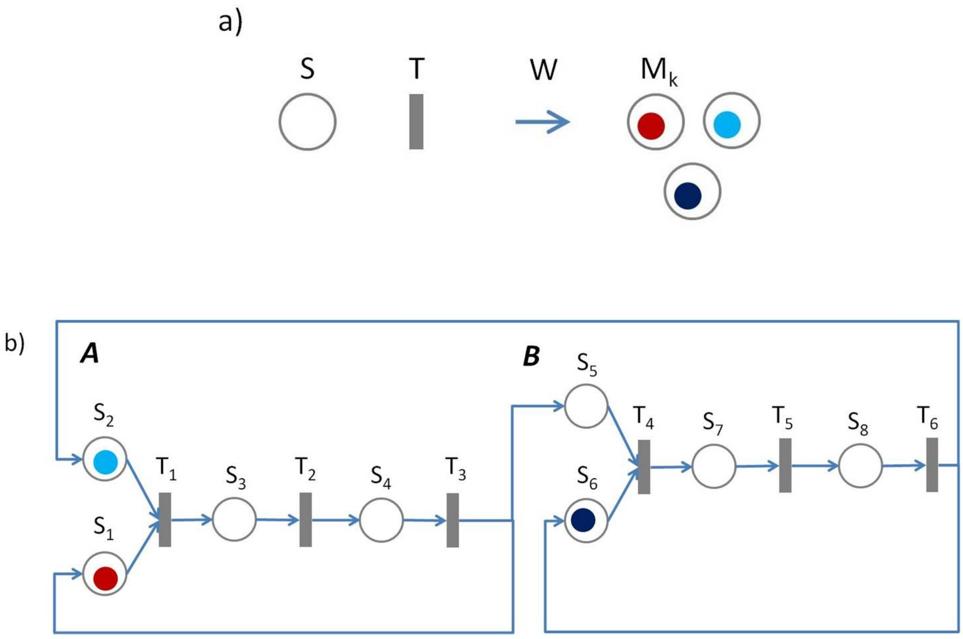


Figure 5.1: A) Graphical components of Petri Nets B) An arbitrary (Simple Mark) Petri Nets model representing the scenario of two interacting agents.

becomes $M_1 = \{s_3, s_6\}$, which in turn enables t_2 for the next iteration (not shown in the figure). t_4 is fired as soon as a token reaches s_5.

5.3 Design and Implementations

As mentioned previously the main aims in this work are: 1) to show how arbitrary discrete-event systems can be designed by DNA-based reactions, and 2) to show how modeling control architecture of distributed agents can implement such systems. We evaluate the first objective via wet-lab experiments, and the second one via in-silico simulation. We were unable to completely implement the second objective in vitro due to the limitation of current available technology in single molecule DNA preparation in a test tube.

We formalize a bottom-up approach of three different steps of implementation. First, we define the low-level reaction controllers. DNA strand displacement is employed as the main mechanism for reaction dynamics. This molecular operation is solely based on competitive hybridization between strands that share common binding regions with other strands in a partially annealed complex; as it does not require the additional design of different molecules, such as restriction enzymes, it is possible to construct a modular design for large-scale application (1, 13). The basic reaction of DNA strand displacement was previously explained in Section 2.1.1.

Second, we translate these low-level primitives into a higher-level representation by using Petri Nets. The dynamics of Petri Nets are denoted by the changing of the tokens' position, which represent chemical reactions between DNA strands. As in DNA strand displacement, the main fuel for the reaction is the partially pre-hybridized DNA complexes (or DNA gates), and we thus model them as the set of transitions T. The set of places P is represented by DNA signals that are unique for each place. The marking state M_k is a subset of signals that are active at particular time k. It is also important to note that the design of DNA gates is mainly dictated by an arc connection W that imposes signals that a transition can consume and produce. The nucleotide assignments for each DNA signal can be performed by following a rule of thumb (52, 96), assisted by some simulation software to minimize crosstalk and secondary structures (58) (refer to Section 5.4.1 for further description).

Generally, a complete DNA-based Petri Nets model can be built from three basic transitional operations.

1. *sequential*, is an operation that converts one input signal to one output signal. It can be written as

$$\mathrm{S}_1 \xrightarrow{T_1} \mathrm{S}_2$$

   which means that by the composition of a signal S_1 with a gate T_1, a reaction occurs that transforms S_1 into a new signal S_2.

2. *synchronization*, is an operation that converts two or more input signals into one output signal. It can be written as

$$\mathrm{S}_1 + ... + \mathrm{S}_\mathrm{m} \xrightarrow{T_1} \mathrm{S}_\mathrm{n}$$

which means that by the composition of signals $S_1, ..., S_m$, with a gate T_1, a reaction occurs that transforms all input signal into a new signal S_n.

3. *concurrent*, is an operation that converts an input signal into two or more output signals. It can be written as

$$\mathrm{S_1} \xrightarrow{T_1} \mathrm{S_2} + ... + \mathrm{S_n}$$

which means that by the composition of a signal S_1 with a gate T_1, a reaction occurs that transforms S_1 into new signals $S_2, ..., S_n$.

Figure 5.2 summarizes the implementation of DNA-based reactions for each of these operators, in addition to their Petri Nets equivalent mapping, while Figure 5.3 shows the initial experimental results. Details of reaction of pathways for each operator is omitted, please refer to Section 5.4.2. The horizontal axis captures the experiment steps, each mark shows the injection of the gate and input strands in order. The vertical axis measures the normalized fluorescence intensity. 0 values are determined in two different ways: in case all leaks are considered (leaky), and in case all leaks are neglected (ideal).

A place in Petri Nets is represented by a single strand input, and a transition is represented by a multi-strand gate complex that accepts corresponding inputs. Another intermediate single strand DNA is required to trigger the output once all inputs are presented, otherwise it reversibly releases the inputs. The initial states are on the left side, and the final states in the middle of the figure. The complete reaction pathway of each operator implementation is shown in Section 5.4.2. Kinetics experiment results are shown in the right-side of the figure. The horizontal axis captures every step of the experiment, from the injection of the gate complex (gray circle) to until all inputs (colored down triangles) are included in the solution. The output level is measured in the end (up triangle). The vertical axis measures the fluorescence intensity changes. These values are normalized to be between 0 and 1. In leaky reactions, 0 values are determined by the time when no DNA strands are present within the solution; whereas in ideal reactions, all leaks are neglected; therefore 0 values are determined by the time when the first input is mixed.

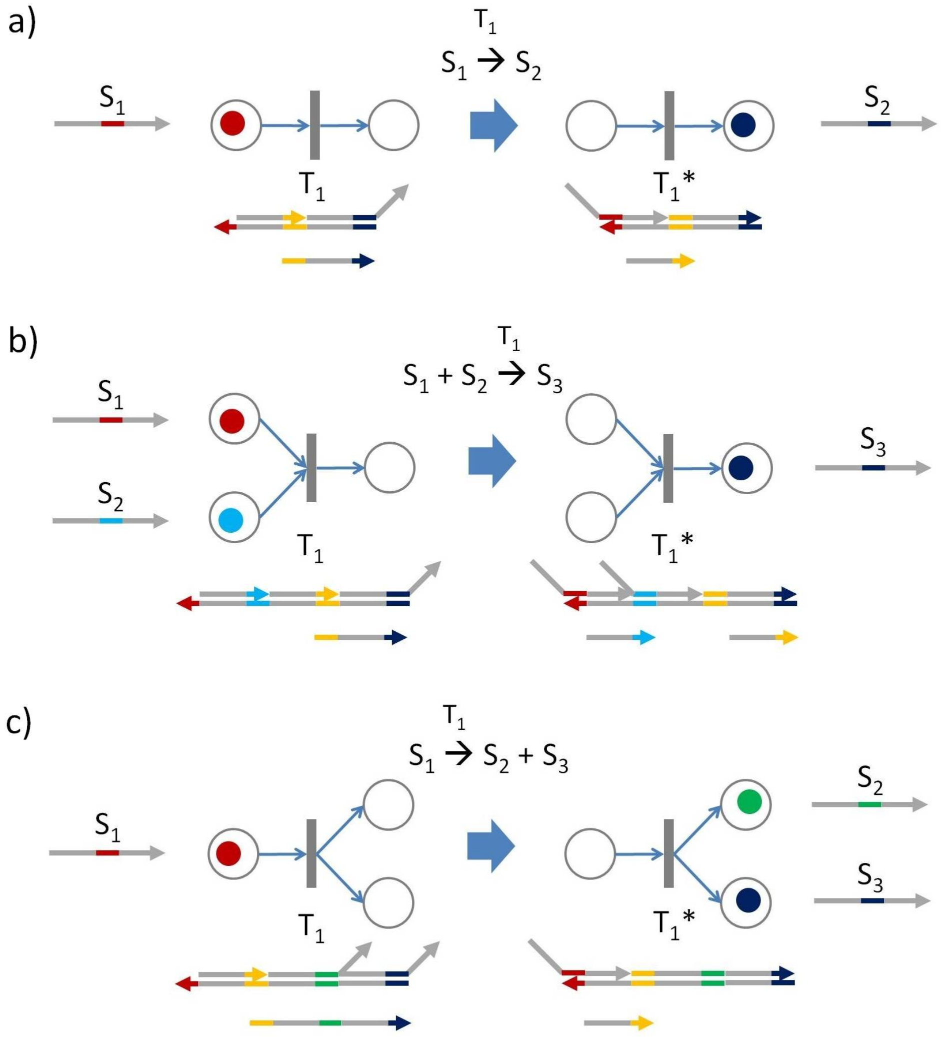


Figure 5.2: Implementation of each Petri Nets operator in a DNA-based reaction. A) sequential operator. B) synchronization operator. C) concurrent operator

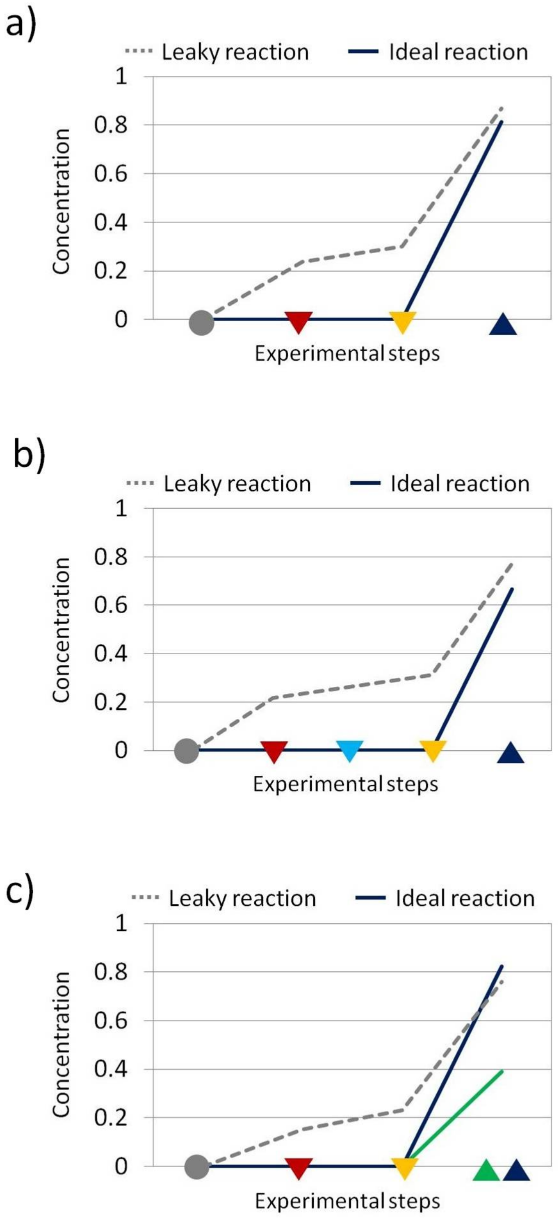


Figure 5.3: Experimental results. A) sequential operator. B) synchronization operator. C) concurrent operator

Leaking in strand displacement reactions is a typical case, in which gate complexes are not formed perfectly during the annealing process, leaving small chances for unintended interactions to happen (1, 13). Thus, there are slight growth in fluorescence measurements even before all inputs are presented.

For all operators, the behaviors are correctly confirmed, i.e. the output strands are only released when all input strands are presented. Raw time-scan data measurements are presented in Section 5.4.3.

The kinetics of each reaction were measured through wet-lab experiments to confirm their behaviors. Complete reaction pathways and time-scan data measurements are provided in Section 5.4.3. Moreover, a general transformation operator can be obtained by combining synchronization and concurrent operators together to implement a chemical equation:

$$\mathrm{X_1} + ... + \mathrm{X_m} \xrightarrow{T} \mathrm{Y_1} + ... + \mathrm{Y_n}.$$

Third, we model the high-level behavior of a DNA-based system into an abstraction that can be accepted by Petri Nets. For a case study, we employ a scenario of two interacting agents, namely agent A and B, performing arbitrary functions in a distributed and coordinative manner. Agent A first performs any given task, while agent B waits for it to finish. As soon as agent A finishes, it sends a signal for agent B to start its own task. Agent B undergoes the same process, and in turn send back a notification for agent A that starts the next iteration. This task scheduling process can be represented in a Petri Nets model as depicted by Figure 5.1.b, hence the simulation in DNA-only reactions. DNA strands are used to represent their action states, and the strand displacement reaction is used to represent the signaling process for state transitions. These definitions are equivalent to the set of places and transitions in Petri Nets.

Figure 5.4 shows the simulation results of the interaction model of the DNA-based agent system. The model is a closed-loop design in which the system continues working and does not end in a particular state for as long as the fuel is provided. In this figure, we show two cycles of operations, separated by the bold dashed line. Each species represents a single copy of different DNA strands; a single-strand DNA represents a place in a Petri Nets (their correspondence is color-coded), and a multi-strand complex represents a transition. Each line on the

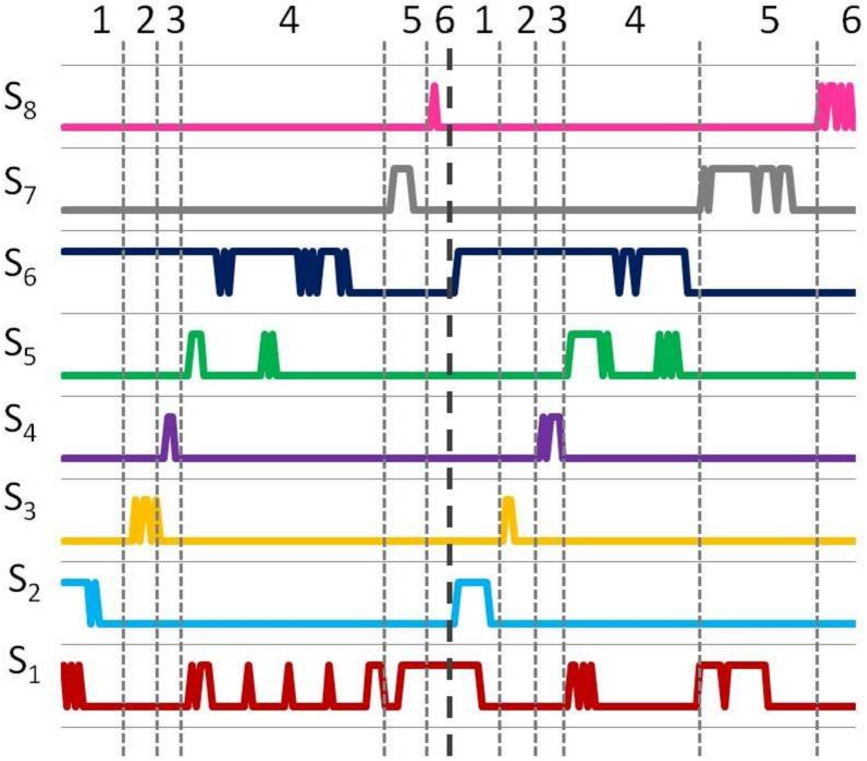


Figure 5.4: Stochastic simulation result of a distributed DNA-based agent system.

graph depicts the activeness of each DNA strand (only sets of places are shown). A high bar means active, and a low bar means inactive (or partially inactive because of reversible binding). The strand displacement and branch migration rates are assumed to be uniform (approximately 1.126×10^{-1}/s and 3×10^{-4} /nM/s), so that the kinetics difference and delay caused by nucleotides design of all strands can be minimized.

Corresponding Petri Nets states are shown in numbered figures in Figure 5.5, and their occurrences are indexed in Figure 5.4. Each state is separated by thin dashed lines. For example, in state=1, all strands are in the low bar, and hence inactive, except for S_1, S_2, and S_6. This condition corresponds to the existence of a token at places S_1, S_2, and S_6. The correctness of the remaining states can be inferred in the same way. Note that, a strand can occasionally be reversibly consumed, but it cannot proceed to a further step, for example strand S_1 at state=4, cycle=1. There are many fluctuations that make the token temporarily disappear. However, to proceed further strand S_2 needs to be present.

5.4 Methods

5.4.1 Sequence Design

The sequence design in this work is done at domain level. Set of sequences was obtained from Visual DSD. They are modified manually in order to satisfy following rules (52):

1. One sequence is made from only three bases: A, C, and T to minimize secondary structure; only complementary sequence to this has G-bases.

2. No more than four consecutive A and T in a row, and no more than three consecutive C in a row to reduce synthesis error.

3. The number of C-bases is in between 30%-70% to ensure comparable melting temperature.

After being generated, all sequences are checked by using NUPACK (58), and are manually modied if necessary.

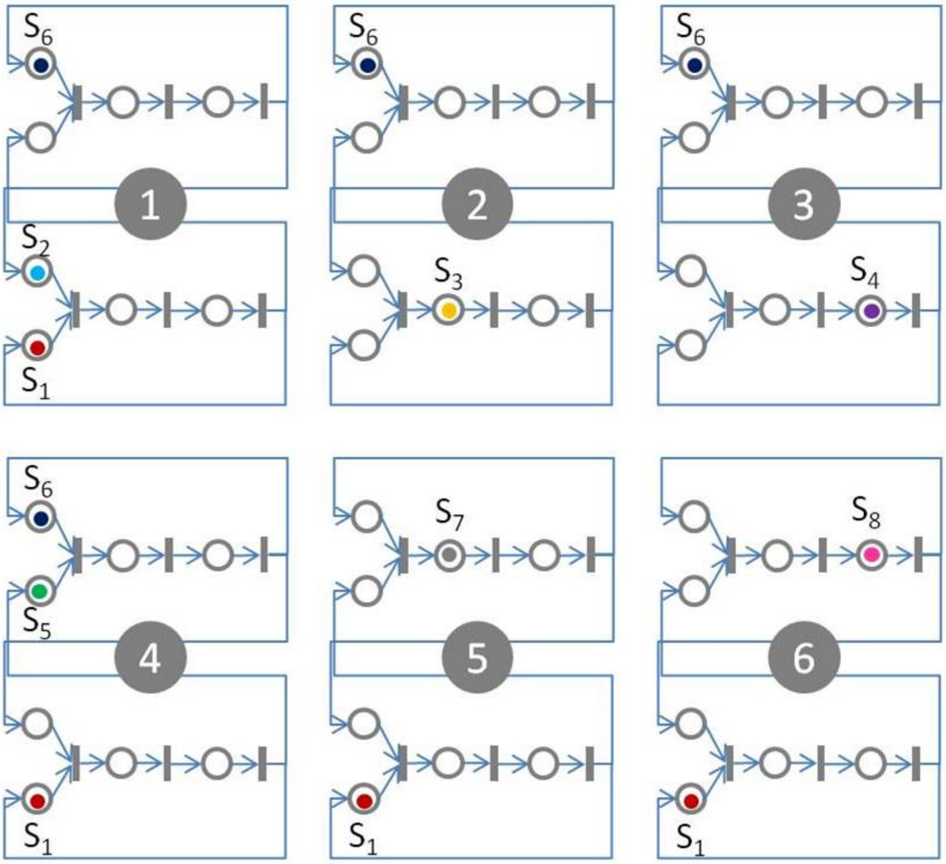


Figure 5.5: Corresponding Petri Nets states to the simulation result.

5.4.2 Reaction Pathway

The following figures (Figure 5.6 - 5.8)are detailed reaction pathways for each Petri Net operator as experimental results are summarized in Figure 5.2. Images are obtained from graphical simulations of Visual DSD. Bold-line species are provided as reactants, arrows show reaction direction, where uncolored means forward reaction and black-colored means backward reaction. Alphanumeric characters represent DNA sequence coding which correspond to Table 5.1.

Table 5.1: List of DNA sequences

No	Domain	Sequence
1	$< yb \quad t >$	CCAATTTCTAACCTAAACAACTCAAT
2	$< xh \quad xt \quad xb >$	CCTTCCACTTAACACATCTCCCTTATCCATTTACATTACATAACCA
3	$< xt^* \quad xh^* \quad t^* \quad yb^* \quad yt^* >$	ATAAGGGAGATGTGTTAAGTGGAAGGATTGAGTTGTTTAGGTTAGAAATTGGTTGGTA
4	$< yh \quad yt \quad yb >$	CCTACTACATAACCACCAACTACCAACCAATTTCTAACCTAAACAA
5	$< t \quad xh \quad xt >$	CTCAATCCTTCCACTTAACACATCTCCCTTAT
6	$< ab \quad zt >$	CCATCATCATATCCATACTCTATCCA
7	$< zb \quad t >$	CCTTAAACAAACCAATCCAACTCAAT
8	$< xt^* \quad xh^* \quad t^* \quad zb^* \quad zt^* \quad ab^* \quad at^* >$	ATAAGGGAGATGTGTTAAGTGGAAGGATTGAGTTGGATTGGTTTGTTTAAGGTGGATAGAGTATGGATATGATGATGGTGAGTG
9	$< ah \quad at \quad ab >$	CTCCTTACTACACTCCACACCACTCACCATCATCATATCCATACTC
10	$< zh \quad zt \quad zb >$	CACTCTACACCACATACCACTATCCACCTTAAACAAACCAATCCAA
11	$< t \quad yh \quad yt \quad xh \quad xt >$	CTCAATCCTACTACATAACCACCAACTACCAACCTTCCACTTAACACATCTCCCTTAT
12	$< xt^* \quad xh^* \quad yt^* \quad yh^* \quad t^* \quad zb^* \quad zt^* >$	ATAAGGGAGATGTGTTAAGTGGAAGGTTGGTAGTTGGTGGTTATGTAGTAGGATTGAGTTGGATTGGTTTGTTTAAGGTGGATA
13	$< xt \quad xb >$	[6-FAM]TGGTTATGTAATGTAAATGGATAAGG
14	$< xb >$	CCATTTACATTACATAACCA[Dabcyl-Q]
15	$< yt \quad yb >$	[AminoC6+ROX]TTGTTTAGGTTAGAAATTGGTTGGTA
16	$< yb >$	CCAATTTCTAACCTAAACAA[BHQ2a-Q]

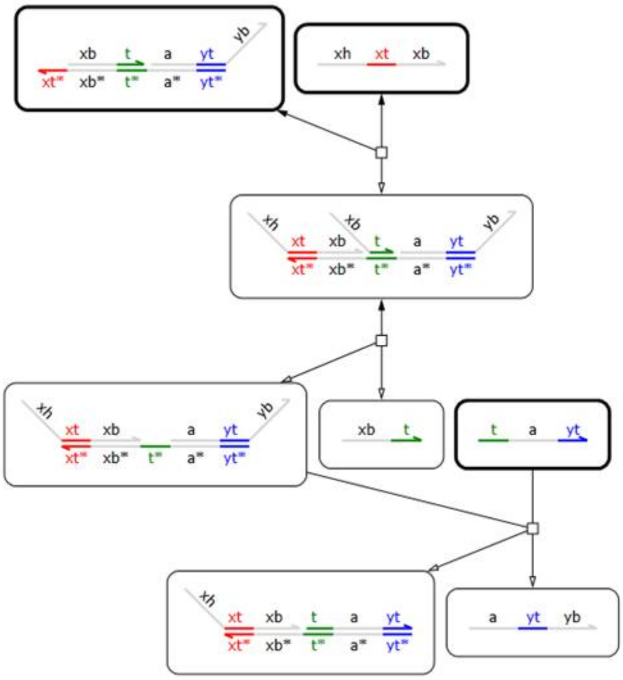


Figure 5.6: Sequential operator reaction pathway.

5.4.3 Experimental Results

Figure 5.9 shows raw time-scan measurement data for sequential operator. FAM fluorescence is used as the reporter molecule (EX=495/EM=520). Vertical axis is in arbitrary unit (might vary from time to time depending on fluorescence type, simulation, and machine condition), and horizontal axis is time in seconds (t=3000). Other parameters are: delay=5s, EX/EM slit=5.0, PM voltage=700.

Figure 5.10 shows raw time-scan measurement for synchronization operator. FAM fluorescence is used as the reporter molecule (EX=495/EM=520). Vertical axis is in arbitrary unit (might vary from time to time depending on fluorescence type, simulation, and machine condition), and horizontal axis is time in seconds (t=3000). Other parameters are: delay=5s, EX/EM slit=5.0, PM voltage=700.

Figure S.7: Raw time-scan measurement for concurrent operator. FAM and ROX fluorescence are used as reporter molecules (EX=495/EM=520 for FAM, (EX=580/EM=610) for ROX). However, in this experiment, only FAM channel is selected for measurement. Vertical axis is in arbitrary unit (might vary from

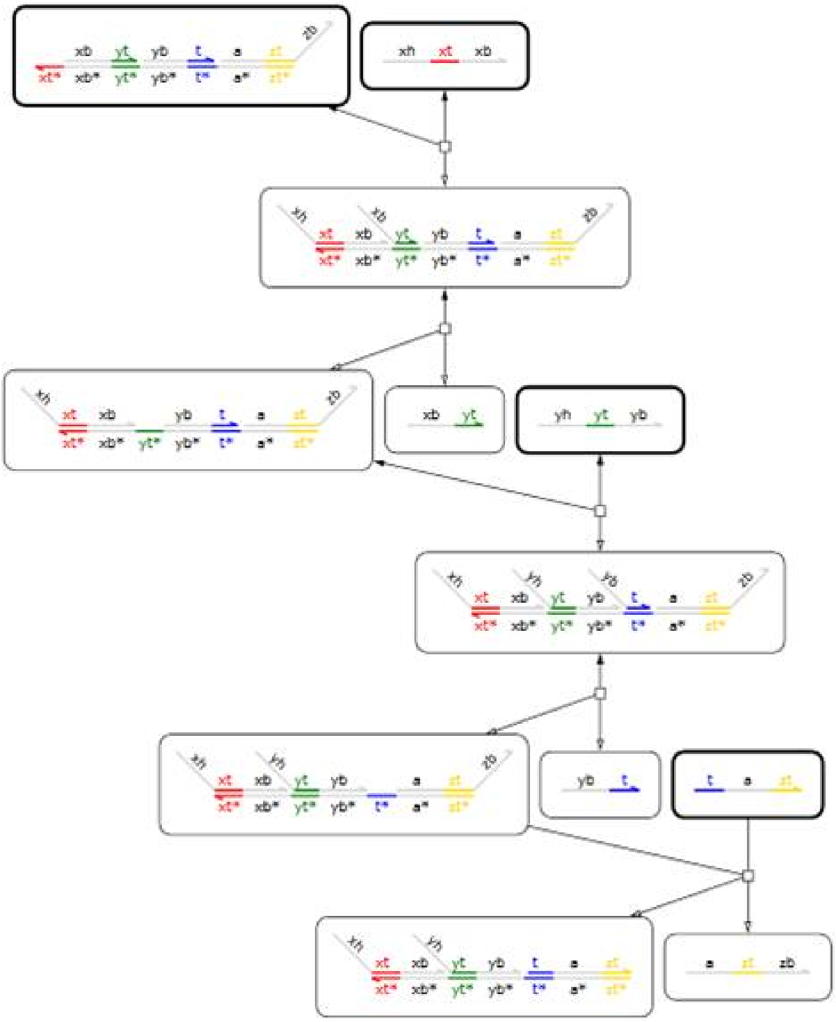


Figure 5.7: Synchronization operator reaction pathway.

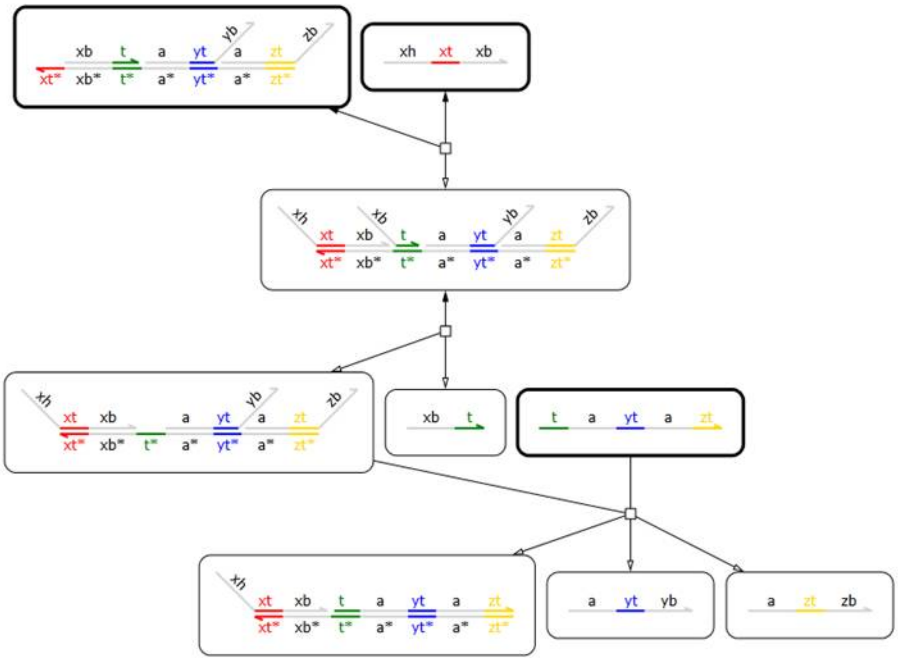


Figure 5.8: Concurrent operator reaction pathway.

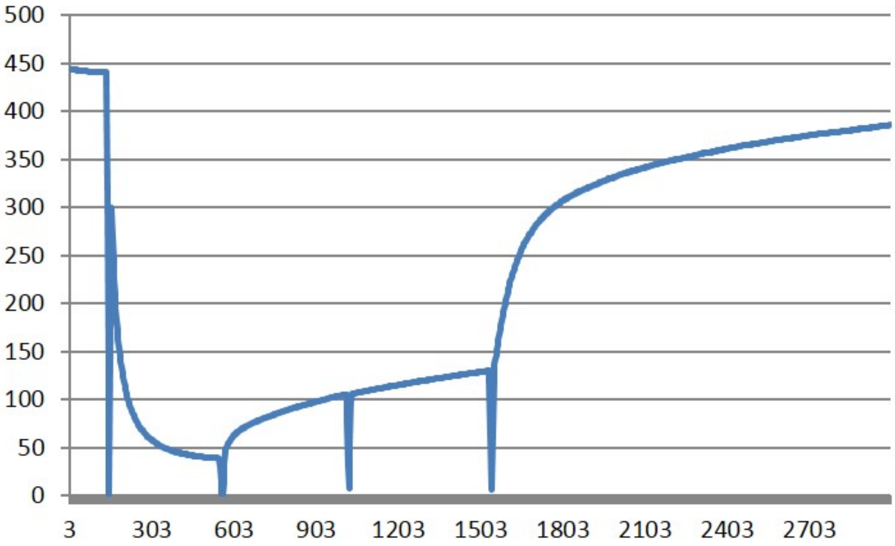


Figure 5.9: Sequential operator.

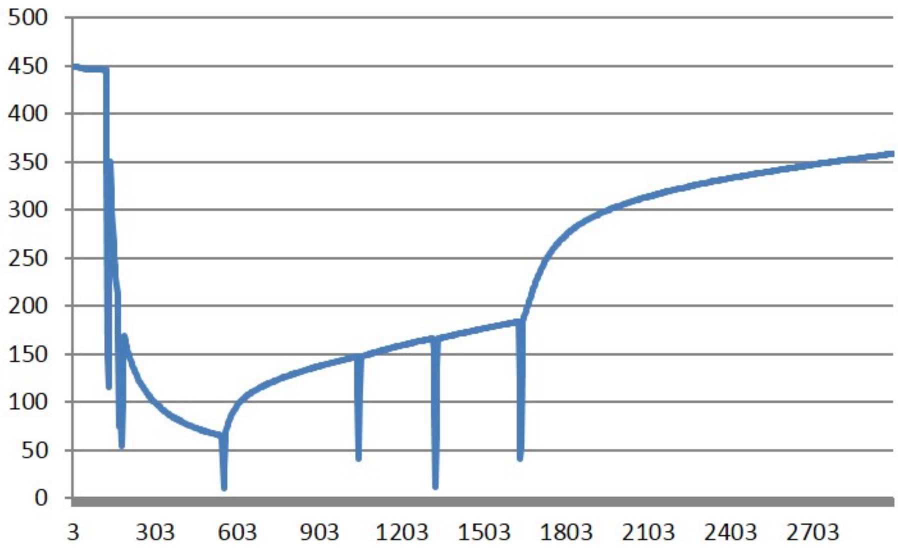


Figure 5.10: Synchronization operator.

time to time depending on fluorescence type, simulation, and machine condition), and horizontal axis is time in seconds (t=3000). Other parameters are: delay=5s, EX/EM slit=5.0, PM voltage=700.

Figure S.8: Raw wavelength measurements for concurrent operator (because we were not able to measure both fluorescence channels at the same time, due to machine limitation). Vertical axis is in arbitrary unit (might vary from time to time depending on fluorescence type, simulation, and machine condition), and horizontal axis is within the range of emission wavelength of respective fluorescence molecules. a) FAM channel, b) ROX channel.

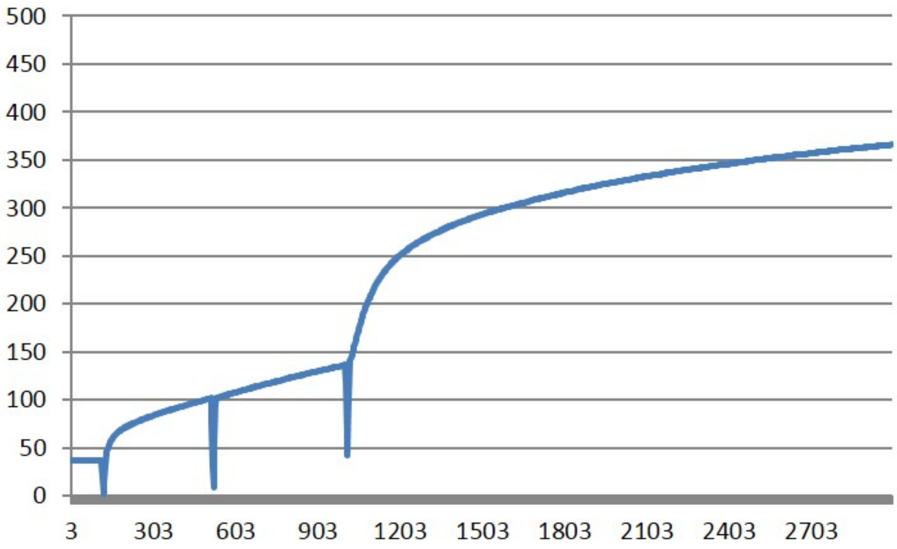


Figure 5.11: Concurrent operator.

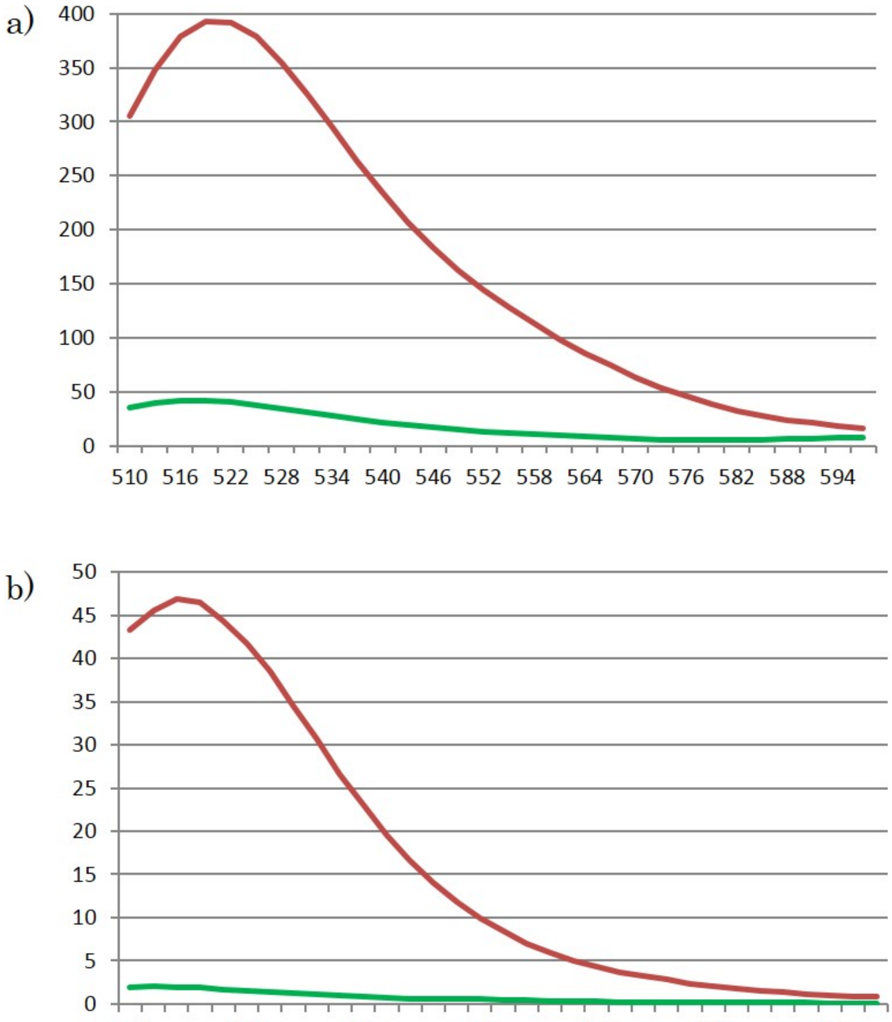


Figure 5.12: Concurrent operator (wavelength).

6

Probabilistic DNA Gate

Molecular robotics and autonomous molecular machines, like their conventional-mechanical counterparts, are expected to perform intelligent tasks under minimum external supervisions. One strategy to accomplish such complex design is to represent internal states of the machines by using finite state automaton (or also called finite state machine/FSM). The transition between states is triggered by external stimulus (can be input signals, sensing data from the environment, or the communication with other machines in the case of multi-agent systems). While there have been many proposals in implementing deterministic transitions by DNA reactions, e.g. by DNA strand displacement cascades, the experimental procedure still remains a challenge. Moreover, in this work, we also propose a new design for stochastic transitions, which also may be applied to arbitrary stochastic DNA computations.

6.1 Introduction

To date, one important challenge in DNA nanotechnology is to design intricate information processing mechanisms that allow the development of autonomous and programmable systems based on biomolecular reactions (25, 26). One promising approach is by employing the isothermal competitive hybridization mechanism, termed DNA strand displacement (DSD), to achieve complex dynamical functions at the molecular level (10, 12, 13, 51).

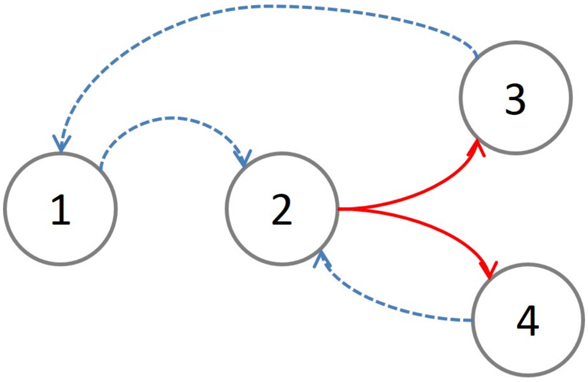


Figure 6.1: Finite state machine or FSM

The major obstacle is, encapsulating computation through chemical reactions is not as trivial as in silicon-based circuitry. Most of efforts rely on engineering by reaction kinetics that is, in principle, based on continuous model of ordinary differential equations (9, 10, 53). Discrete systems, however, are easier to understand and provide elegant mechanism for sequential logic programming (82). In this work, we strive to contribute to solve this challenge by surveying an experimental implementation of DNA-based systems based on discrete models, such as Finite State Machine.

Finite State Machine (FSM), or Finite State Automaton, is an abstract computational model used to describe discrete systems, including programming and logical sequences (82). For long, FSM has been used to model various complex design tasks, including autonomous and intelligent robotics. In this paper, rather than discussing the FSM by its formal definition, we represent it with the graphical notation.

Generally, a robot, either mechanical or biomolecular ones, can be represented as an abstract model consisting sequences of internal states. These capture possible functions the robot may perform, as well as how it progresses from one state to another, through the transitioning process. As shown in Figure 6.1, a FSM consists of states (denoted by numbered circles) and transitions (denoted by dashed and solid lines). One state may move to another while certain required conditions or events are satisfied, e.g. external stimuli or time evolution. These transitions may be exclusive to one possible outcome (deterministic, represented

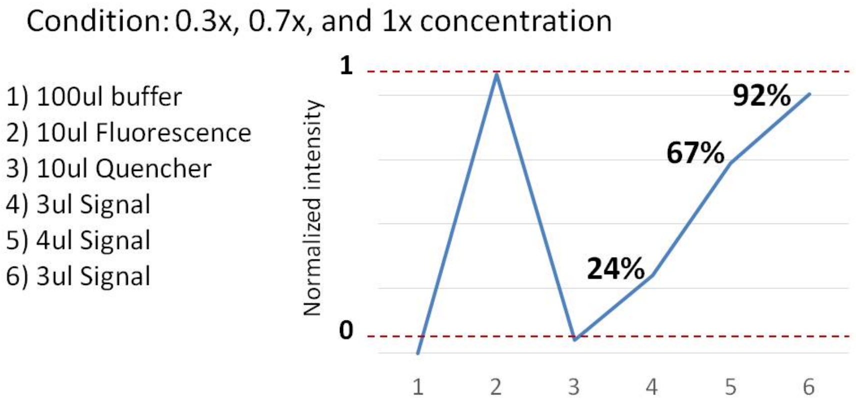


Figure 6.2: Experimental result of the reporter test

by dashed-blue lines), or with possibility of multiple outcomes (stochastic, represented by solid-red line). In this work, we show how a FSM can be designed over bio-molecular reactions, by employing DNA strand displacement reactions.

6.2 Design and Results

DSD can be cascaded into multilayer networks, where output signals from one reaction serve as input signals for another reaction. Moreover, one signal may serve as inputs in two different reactions (OR gate design), or multiple inputs should be present for a reaction to occur (AND gate design). Thus, in principle, it is possible to build discrete-finite systems based on DNA reactions, such as Petri Nets (11, 60, 64).

In the present study, we survey the implementation of the FSM based on this biomolecular mechanism. To simulate the functionality of the FSM, we experimentally tested four different setups: 1) reporter test, 2) deterministic transition, 3) conditional transition, and 4) stochastic transition.

The first mentioned is the quantitative measurement method of DSD reactions by using FRET technique. The design of the reporter is a slight modification of the basic DSD of Figure 2.1. A pair of DNA strand is labeled with fluorophore (lower strand) and quencher molecules (upper strand), which are complement to each other. When they hybridize, the quencher absorbs the light emitted by the fluorophore, thus no emission is measured. Once an output signal that can displace the top strand of the reporter gate, the quencher will be separated from the fluorophore, resulting in light emission. The growth of the fluorescence intensity is linear to the number of the displacing output signal, thus this may be utilized as unit for signal's measurement. As this value may vary depending on experimental and machine conditions, typically, post-experimental normalization is needed, to map the measured values from 0.0 to 1.0. We utilize two different fluorophores, FAM (excitation 495nm, emission 520nm) and ROX (excitation 580nm, emission 610nm). Results from reporter tests are shown in Figure 6.2. We tested different amount of signals by increasing from 0.3x to 0.7x to 1.0x concentration, and obtained corresponding results with the expected probability.

The second mentioned can be implemented by a simple translator gate that transforms one signal strand into another. In order to build a composable system, the signal's coding structure should be uniform. In this paper, we utilize 3-domain single strand DNA structure: $(non-toehold, toehold, non-toehold)$ which is inspired by (64). Consequently, this determines the design of the gates required for reactions to occur. Figure 6.3 shows the design of DSD reaction that can take one input signal and to transform it into another output signal. One additional single strand acts as a helper strand to ensure that the final step in the reaction is irreversible for a deterministic operation (it possesses different structure with the signals). All reactants are represented by bold lines, intermediate states and outputs are by thin lines. White arrows represent forward reactions, and black arrows represent backward reactions.

The reaction begins with the interaction of toehold x_t from signal X, as an input, with its complement on gate T_1. This is followed by the displacement of non-toehold x_b that it shares with the gate. This step reversibly releases an intermediate single strand due the unstable binding of toehold t. As a new free toehold is now exposed, the helper strand can react with the gate and undergo the similar process to release signal Y, as an output. In addition to that, the gate will also turn into a waste as there is no toehold left open anymore. These whole processes can be written simply as the transformation of signal X into Y which is ruled by gate T_1. Experimental result of deterministic transition is shown in Figure 6.4.

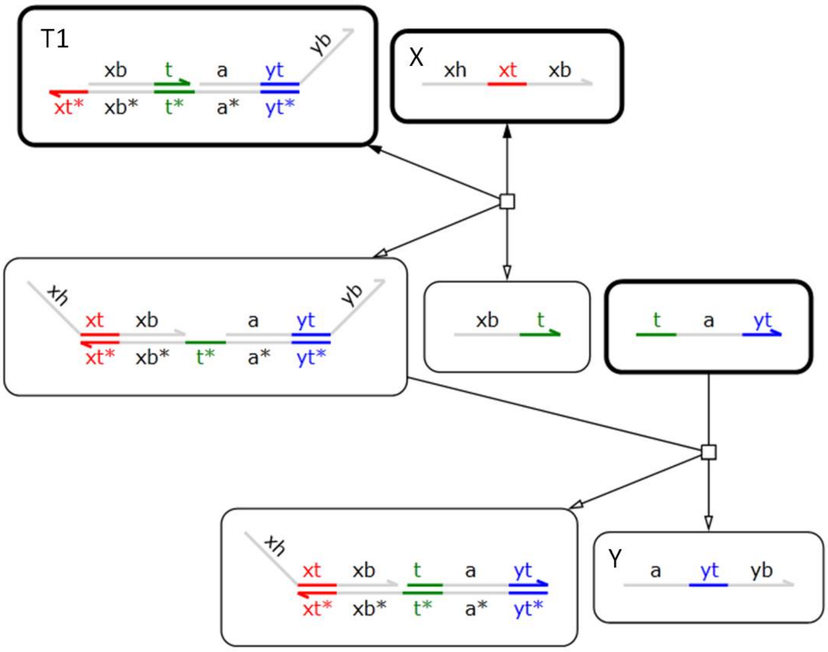


Figure 6.3: Schematic of deterministic transition

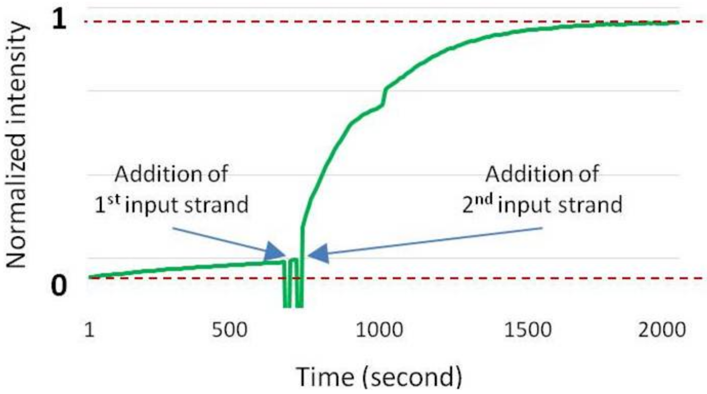


Figure 6.4: Experimental result of deterministic transition

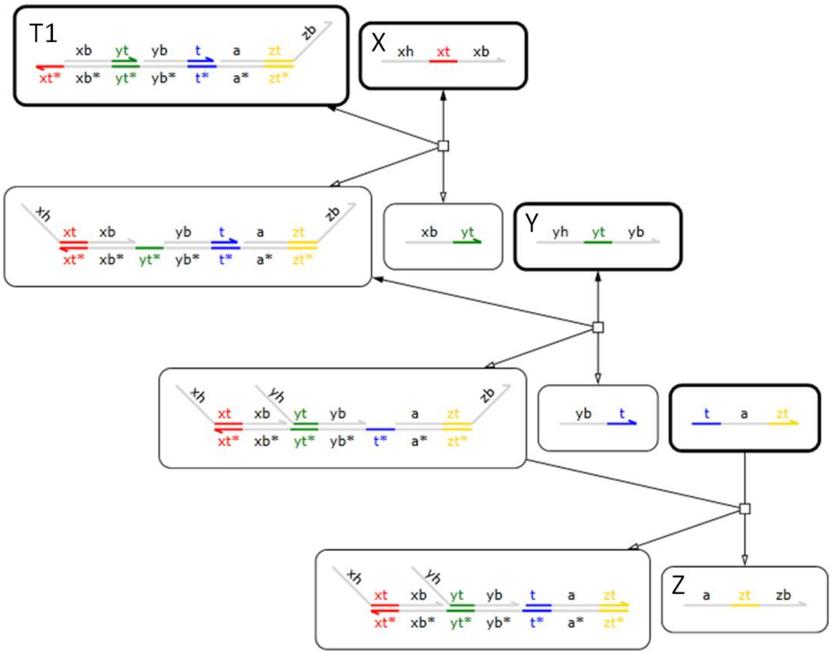
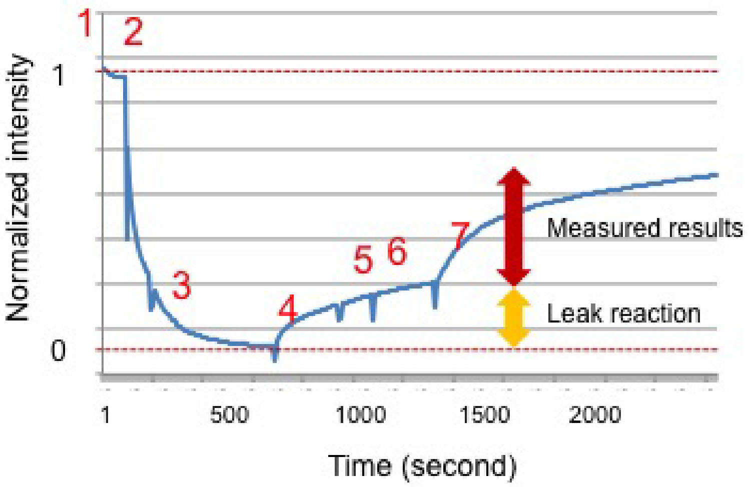


Figure 6.5: Schematic of conditional transition

The third mentioned can be implemented by designing extension reactions that take multiple input signals and to transform them into one output signal, by adding extra domains in the gate. Therefore, there will be additional intermediate reactions depending on the number of input signals. After the last input signal binds, similar to the deterministic transition, the helper strand plays its role to release the output signal, which makes the whole reaction irreversible. Figure 6.5 shows the extension design for the conditional transition (assuming a ”condition” happens once two input signals are present). All reactants are represented by bold lines, intermediate states and outputs are by thin lines. White arrows represent forward reactions, and black arrows represent backward reactions.

The reaction begins with the interaction of toehold xt from signal X, as the first input, with its complement on gate T_1. This is followed by the displacement of non-toehold xb from the gate, which reversibly releases an intermediate single strand. In contrast to the deterministic transition, this step is repeated once again for signal Y, as the second input, until toehold t is exposed. Finally, the helper strand reacts with the gate and the output signal Z is released. After



Experimental procedure

1) 550ul buffer (1x TAE 12.5mg+), 5ul FAM
2) 5ul Quencher
3) 2ul Quencher
4) 125ul Gate
5) 5ul First Input (1st signal input)
6) 5ul Second Input (2nd signal input)
7) 7ul Third Input (trigger)

Figure 6.6: Experimental result of conditional transition

the reaction, gate T_1 turns into a waste as well. It is important to note that by this design, the DSD reaction may not proceed unless both input signals are present. When the second input is missing, the reaction can be undone through the reversible reaction of the first intermediate strand. Then, the first input signal can be released again into the system. Experimental result of conditional transition is shown in Figure 6.6.

The conditional transition is an important concept to build a stochastic-based process, which is utilized to implement the last mentioned. Conceptually, it consists of two conditional transitions with a shared second input signal. The first input signal will be the control of the probability of both reactions.

As shown in Figure 6.7, the stochastic transition is utilized to transform signal Z into two possible outcomes: signal X and signal Y. In this case, signal Z acts as the shared input between gate T_1 and T_2. Both of gates have different first input: signal A and signal B respectively (we refer them as implicit signals as

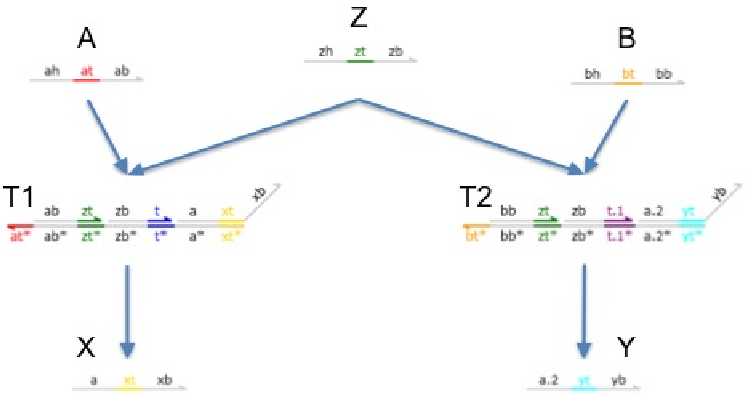
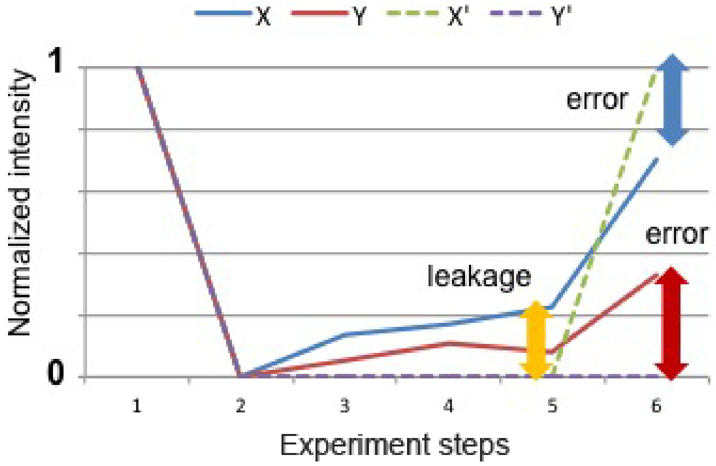


Figure 6.7: Schematic of stochastic transition

they do not directly represent any state within the designed system). Assuming that initial concentrations of signal A and B, and gate T_1 and T_2 are the same (DSD reactions also at the same biochemical rates), the consumption rate of both gates will be uniform as well. Given any arbitrary concentration of signal Z, it will then be equally divided by both gate T_1 and T_2 when the second step of reactions becomes available. This yields 50:50 production of signal X and signal Y.

In the case of different amount of implicit signal's concentration (however both gates are available in arbitrary larger concentrations), the consumption rate of gates depends on the implicit signal's concentration respectively. For example, if signal A is twice larger than signal B, gate T_1 will bind twice more than gate T_2. Because the whole DSD reaction and branch migration are occurred by chance, and more gate T_1 are now available, signal Z binds more likely with gate T_1 rather than gate T_2. The output production will be roughly 67:33 yield. Since these output signals can be cascaded into input signals for different gates, we can safely assume that the reaction probability of a stochastic DNA operator is directly proportional to the implicit strands.

Experimental result of the stochastic transition is shown in Figure 6.8. From this result, one output strand grow faster than another because of external stimulation from the implicit strand. However, in contrast to the ideal outcome (dashed



Experimental procedure

1) 500ul buffer (1x TAE 12.5mg+), 5ul Fluorescence
2) 6ul Quencher
3) 105ul Gate
4) 5ul First Input (1st signal input A)
5) 5ul Second Input (2nd signal input Z)
6) 6ul Third Input (trigger: both)

Figure 6.8: Experimental result of stochastic transition

line), there are errors observed presumably due to reaction leakage. As discussed in previous chapter, leaking in strand displacement reactions is quite a typical case. One possible cause is because the imperfect formation of gate complexes, leaving small chances for unintended interactions to happen. While this aspect has been a major drawback of DNA strand displacement reactions so far, some works reported that leakage can be suppressed by doing purification the DNA samples before further experiments.

All DNA sequences were designed by following the rules as outlined at Chapter 2.2.1. Table 6.1 shows all strands used in these experiments.

Table 6.1: List of DNA sequences

No	Sequence	Transition
1	CCAATTTCTAACCTAAACAACTCAAT	D
2	CCTTCCACTTAACACATCTCCCTTATCCATTTACATTACATAACCA	D, C, S
3	ATAAGGGAGATGTGTTAAGTGGAAGGATTGAGTTGTTTAGGTTAG AAATTGGTTGGTA	D
4	CCTACTACATAACCACCAACTACCAACCAATTTCTAACCTAAACAA	D
5	CTCAATCCTTCCACTTAACACATCTCCCTTAT	D, C, S
6	CCATCATCATATCCATACTCTATCCA	C, S
7	CCTTAAACAAACCAATCCAACTCAAT	C, S
8	ATAAGGGAGATGTGTTAAGTGGAAGGATTGAGTTGGATTGGTTTGTTTAAGGTGGATAGAGTATGGATATGATGATGGTGAGTG	C, S
9	CTCCTTACTACACTCCACACCACTCACCATCATCATATCCATACTC	C, S
10	CACTCTACACCACATACCACTATCCACCTTAAACAAACCAATCCAA	C, S
11	CATCCTCATTACAATCCATCTATCCA	S
12	TTGGTAGTTGGTGGTTATGTAGTAGGATTGAGTTGGATTGGTTTGTTTAAGGTGGATAGATGGATTGTAATGAGGATGTTGGAA	S
13	CTACACTCAATTCACCTCAATTCCAACATCCTCATTACAATCCATC	S
14	CTCAATCCTACTACATAACCACCAACTACCAA	S
15	[6-FAM]TGGTTATGTAATGTAAATGGATAAGG	R
16	CCATTTACATTACATAACCA[Dabcyl-Q]	R
17	[AminoC6+ROX]TTGTTTAGGTTAGAAATTGGTTGGTA	R
18	CCAATTTCTAACCTAAACAA[BHQ2a-Q]	R

7

Conclusions

7.1 Summary

This thesis is a summary of four independent but related research papers on computational design of synthetic bio-molecular systems based on DNA strand displacement reaction.

In the first part, the design of the DNA-based circuit in well-mixed chemical systems is reported. As the characters of DNA-based systems are in many ways similar to many self-organizing models in the real life, a nature-inspired computation has been treated as a promising strategy to compromise the difficulties to deliver computation above bio-molecular materials. In this work, we successfully implemented a nature-inspired self-organizing algorithm, namely Immune Network Theory, by using DNA strand displacement (DSD) based operator. We first expressed the intended behavior of our model in terms of chemical reaction systems and derived its mathematical formulation by Chemical Reaction Networks (CRNs). From its chemical notation, the required DNA operators were designed. We then compared the simulation-based implementation of this DNA-based system to the direct programming of the Immune Network Theory's equation by silicon-based computing. From the obtained results, there is positive correlation can be concluded between these two different methods of implementation. This finding is expected to contribute the efforts in overcoming the obstacle in DNA-based molecular programming, which is to find a formal method to compile arbitrary mathematical model into biochemical reactions directly.

In the second part, Ant Double-Bridge System is chosen as an inspiration for designing the spatially localized DNA architecture based on computation with molecular walkers. Many collective and distributed systems in nature exhibit a global level of intelligence driven by local interaction between participating individuals One particular example is ant foraging behavior. This nature-inspired computation has been modeled and implemented for various optimization, machine learning, and robotics-related problems; and now we showed how this algorithm can be applied to the DNA strand displacement reaction. The design can be implemented on DNA nanostructures, with spatially localized architecture. The computation is carried out by synthesized walkers traversing the landscape. Further applications of this computational model include the design of machines capable of decision making, and systems that can learn to adapt to changes in the environment. We believe this is an important feature for the development of fully autonomous, programmable, and evolvable DNA machine in the future.

In the third part, we discuss the model-based coordination strategy for DNA-based agents based on Petri Nets. By treating DNA strands as individuals or agents participating in a massive interaction-based swarm-system, we showed that coordination between two DNA-based agents could be realized. Agent-based modeling has advantages over conventional/non-agent models, because it enables simple and modular design and control. In addition, the architecture can be easily extended for both homogenous and heterogeneous systems. Our approach decomposes the problem into three different layers. First, we described the low-level reaction primitives based on branch migration and DNA strand displacement. Second, we mapped the intermediate-level representation by using Petri Nets. We confirmed the behavior of our architecture by $in-vitro$ evaluation. Last, we employed a simple coordination task between two agents as a test scenario for the DNA-based interaction model. According to the simulation results, we inferred that concurrency control could be achieved through Petri Nets modeling. We argue that our design strategy is feasible for future DNA-based applications, including DNA-based machines and molecular robotics.

In the last part, the in-vitro implementation of the DNA-based Finite State Machine is presented. Furthermore, a design of probabilistic DNA gate is shown, to exhibit stochastic-like computation based on bio-molecular reactions. With recent advances in DNA nanotechnology, DNA-based machines and robotics, are expected to work under minimum external supervisions. Finite State Machine is a prominent model that can represent the abstraction of the molecular machines through a discrete state-transition diagram. In this work, we strive to demonstrate the deterministic and stochastic computation models through wet-lab experiments, that can implement arbitrary DNA-based Finite State Machine. To simulate the functionality of the FSM, there are four different schema tested in separate experimental settings: reporter test, deterministic transition, conditional transition, and stochastic transition. The obtained results from all experiment setups positively correspond to the expected behaviors.

All of these independent results are important for contribution to the development of DNA-based information processing systems or computation. The research ultimate goal is to develop systems that can perform logical function that will be deployed for molecular robotics application. Analogous to any mechanical robotics, high-level behaviors of the designed machine can be abstracted by Finite State Machine, as discussed in Chapter 6. Generally, the implementation of the DNA-based FSM consists of two challenges: how to design and implement the deterministic process and how to design and implement stochastic process. We showed how these both processes, once be implemented in bio-molecular reactions, can be utilized for designing more sophisticated systems that can solve complex computation problems, such as decision-making and reinforcement learning. We demonstrated the interaction model of DNA agents that is based on deterministic operations in Chapter 3 and that is based on stochastic operations in Chapter 4. The first mentioned is by taking inspiration from distributed interaction of human's immune system, while the latter mentioned is by learning from the metaphor of indirect communication mediated by the environment of the ant colony system. We further discussed that both designs. are suitable for two different architectures of DNA strand displacement systems. The first is for well-mixed chemical systems, which is the typical case of bio-molecular reactions, and the second is for spatially localized architecture, which integrates to technologies from structural DNA nanotechnology. Furthermore, we argue that our approach improves currently available techniques in DNA-based circuits and computation. Most of approaches so far attempted to re-engineer electronic circuits principle by using biological components. However, synthetic bio-molecular systems are relatively slow, prone to error, and may never compete the speed and effectiveness of silicon-based machines. So instead, we exploit the self-organizing principle underlying the molecular scale and described how formal design of agent-based model can be utilized to achieve such purpose. This is as discussed in Chapter 5.

7.2 Future Works

Apart from the obtained results, we are aware that the work presented in this thesis is far from finished. We would end this manuscript by highlighting some open questions as the future works.

First, the Immune-Network-based DNA circuit. At the current stage, the designed model is still very simple structurally. Our intention is to observe the behavior of the DNA-based implementation and to see the correlation with its conventional programming counterpart. For the next step, a larger scale of Immune Network's model should be put into consideration. Furthermore, the kinetics gap between the simplified chemical reactions and the DNA operators should also be properly observed. An alternative design of DNA operators with less kinetics gap should be designed for this matter.

Second, the Ant-System-based DNA circuit. The utilization of DNA nanostructure technology in combination with the present DNA circuits is expected to broaden the potential applications for this design. This nature-inspired design may also be may be incorporated with other DNA based computation systems, such as tiles assembly for its modular construction, amplifiers, or even in DNA routing system.

Third, the Agent-based DNA circuit. The scenario we utilized for our model was simplified to avoid confusion in understanding how the design works. A more complex scenario that is based on real-life interaction processes and currently available DNA-based functions is important for actually implementing our method. Second, as there is no technology available at this time, to prepare a single molecule of DNA in solution, we are not able to fully evaluate our system $in-vitro$. In our agent-based model approach, each copy of a DNA strand acts as a single agent in a massive-collective system. Therefore, a single molecule observation technique would be highly advantageous for models similar to ours.

Last, the Probabilistic-based DNA gate. Current results have shown the autonomous capability of the designed circuit to selectbetween two transitions which is directed by the implicit signals. Our implementation was based on gate motifs designed in previous works. However, this design still subjects to leakage-error, due to imperfect DNA synthesize. While other works have shown that post-purification protocol may reduce this problem, our motivation in this work is to show the conceptual idea rather than to present the ideal experimental condition. In the future, in addition to the purification steps, new simplified design may also be tested to obtain better result performances.

Appendix

This section presents the programming code from our in-silico implementation by using Microsoft Visual DNA Strand Displacement simulator (59), and Matlab R2011a. The Visual DSD software, its manual and documentation can be obtained from: `http://research.microsoft.com/en-us/projects/dna/`.

Visual DSD Code

1. Sequential / Deterministic Operator.

```
directive duration 50000.0 points 1000
directive scale 100.0
directive plot sum(<_ xt^ xb>); sum(<_ yt^ yb>)

def signal = 100
def gate = 100

def Sig(N, h, t, b) = N * <h t^ b>

def TGat(N, t0, b0, t1, b1) = new t new a
( N * t0^*:[b0 t^]:[a t1^]<b1>
| N * <t^ a t1^>
)

( Sig(signal, xh, xt, xb)
| TGat(gate, xt, xb, yt, yb)
)
```

2. Synchronization / Conditional Operator.

```
directive duration 50000.0 points 1000
directive scale 100.0
directive plot sum(<_ xt^ xb>); sum(<_ yt^ yb>); sum(<_ zt^ zb>)

def signal = 100
def gate = 100

def Sig(N, h, t, b) = N * <h t^ b>

def JGat(N, t1, b1, t2, b2, t3, b3) = new a new b
( N * t1^*:[b1 t2^]:[b2 a^]:[b t3^]<b3>
| N * <a^ b t3^>
)

( Sig(signal, xh, xt, xb)
| Sig(signal, yh, yt, yb)
| JGat(gate, xt, xb, yt, yb, zt, zb)
)
```

3. Concurrent Operator.

```
directive duration 50000.0 points 1000
directive scale 100.0
directive plot sum(<_ xt^ xb>); sum(<_ yt^ yb>); sum(<_ zt^ zb>)

def signal = 100
def gate = 100

def Sig(N, h, t, b) = N * <h t^ b>

def FGat(N, t1, b1, t2, b2, t3, b3) = new a
( N * t1^*:[b1 t2^]<b2>:[a t3^]<b3>
| N * <t2^ a t3^>
)

( Sig(signal, xh, xt, xb)
| FGat(gate, xt, xb, yt, yb, zt, zb)
)
```

4. Stochastic DNA Operator.

```
directive duration 50000.0 points 1000
directive scale 100.0
directive plot sum(<_ xt^ xb>); sum(<_ yt^ yb>); sum(<_ zt^ zb>)

def signal = 100
def gate = 100

def Sig(N, h, t, b) = N * <h t^ b>

def JGat(N, t0, b0, t1, b1, t2, b2) = new t new a
```

```
( N * t0^*:[b0 t1^]:[b1 t^]:[a t2^]<b2>
| N * <t^ a t2^>
)

( Sig(signal, xh, xt, xb)
| Sig(75, ah, at, ab)
| Sig(25, bh, bt, bb)
| JGat(gate, at, ab, xt, xb, yt, yb)
| JGat(gate, bt, bb, xt, xb, zt, zb)
)
```

5. Immune-Network-Based DNA Circuit.

```
directive sample 2000.0 1000
directive plot sum(<_ xt^ xb>); sum(<_ yt^ yb>); sum(<_ zt^ zb>);
sum(<_ at^ ab>); sum(<_ bt^ bb>); sum(<_ ct^ cb>)

def Species(N, h, t, b) = N * <h t^ b>

def Fuel(N, h, t, b) = constant N * <h t^ b>

def kGat(N, t1, b1) = constant N * t1^*:[b1]

def mGat(N, t1, b1, t2, b2, t3, b3, t4, b4, t5, b5) = new t new a
( constant N * t1^*:[b1 t2^]:[b2 t3^]:[b3 t^]:[a t4^]<b4>:[a t5^]<b5>
| constant N * <t^ a t4^ a t5^>
| constant N * <b1 t2^>
| constant N * <b2 t3^> )

( Species(300, xh, xt, xb)
| Species(200, yh, yt, yb)
| Species(200, zh, zt, zb)

| Species(220, ah, at, ab)
| Species(180, bh, bt, bb)
| Species(200, ch, ct, cb)

| Fuel(1586, k1h, k1t, k1b)
| Fuel(4751, k2h, k2t, k2b)
| Fuel(172, k3h, k3t, k3b)
| Fuel(1908, k4h, k4t, k4b)
| Fuel(3827, k5h, k5t, k5b)
| Fuel(3976, k6h, k6t, k6b)

| Fuel(2449, kz1h, kz1t, kz1b)
| Fuel(2228, ky1h, ky1t, ky1b)
| Fuel(3231, kx1h, kx1t, kx1b)

| Fuel(3547, kz2h, kz2t, kz2b)
| Fuel(3774, ky2h, ky2t, ky2b)
| Fuel(1380, kx2h, kx2t, kx2b)

| Fuel(3398, kz3h, kz3t, kz3b)
| Fuel(3276, ky3h, ky3t, ky3b)
| Fuel(813, kx3h, kx3t, kx3b)

| mGat(4000, k3t, k3b, xt, xb, yt, yb, yt, yb, yt, yb)
| mGat(4000, k5t, k5b, xt, xb, zt, zb, zt, zb, zt, zb)
| mGat(4000, k1t, k1b, yt, yb, xt, xb, xt, xb, xt, xb)
| mGat(4000, k6t, k6b, yt, yb, zt, zb, zt, zb, zt, zb)
| mGat(4000, k2t, k2b, zt, zb, xt, xb, xt, xb, xt, xb)
| mGat(4000, k4t, k4b, zt, zb, yt, yb, yt, yb, yt, yb)

| mGat(4000, kx1t, kx1b, at, ab, xt, xb, xt, xb, xt, xb)
| mGat(4000, ky1t, ky1b, at, ab, yt, yb, yt, yb, yt, yb)
| mGat(4000, kz1t, kz1b, at, ab, zt, zb, zt, zb, zt, zb)
| mGat(4000, kx2t, kx2b, bt, bb, xt, xb, xt, xb, xt, xb)
| mGat(4000, ky2t, ky2b, bt, bb, yt, yb, yt, yb, yt, yb)
| mGat(4000, kz2t, kz2b, bt, bb, zt, zb, zt, zb, zt, zb)
| mGat(4000, kx3t, kx3b, ct, cb, xt, xb, xt, xb, xt, xb)
| mGat(4000, ky3t, ky3b, ct, cb, yt, yb, yt, yb, yt, yb)
| mGat(4000, kz3t, kz3b, ct, cb, zt, zb, zt, zb, zt, zb)

| kGat(3, xt, xb)
| kGat(3, yt, yb)
| kGat(3, zt, zb)
)
```

6. Ant-System-Based DNA Circuit.

```
directive duration 500000.0 points 200
directive scale 100.0
directive plot sum(<_ j1t^ j1b>); sum(<_ j2t^ j2b>)

def gatNum = 50
def antNum = 100
def phero1 = 100
def phero2 = 100
def kilNum = 125

def Sig(N, h, t, b) = N * <h t^ b>

def mGat(N, t0, b0, t1, b1, h2, t2, b2, h3, t3, b3, h4, t4, b4) =
( N * t0^*:[b0 t1^]:[b1 t^]:[h2 t2^]<b2>:[h3 t3^]<b3>:[h4 t4^]<b4>
| N * <t^ h2 t2^ h3 t3^ h4 t4^>
)

def kGat(N, t, b) = N * t^*:[b]

( Sig(antNum, aah, aat, aab)
| Sig(phero1, j1h, j1t, j1b)
| Sig(phero2, j2h, j2t, j2b)

| mGat(gatNum, j1t, j1b, aat, aab, xoh, xot, xob, axh, axt, axb, j1h, j1t, j1b)
| mGat(gatNum, xot, xob, axt, axb, yoh, yot, yob, aah, aat, aab, j1h, j1t, j1b)

| mGat(gatNum, j2t, j2b, aat, aab, boh, bot, bob, abh, abt, abb, j2h, j2t, j2b)
| mGat(gatNum, bot, bob, abt, abb, coh, cot, cob, ach, act, acb, j2h, j2t, j2b)
| mGat(gatNum, cot, cob, act, acb, doh, dot, dob, adh, adt, adb, j2h, j2t, j2b)
| mGat(gatNum, dot, dob, adt, adb, eoh, eot, eob, aah, aat, aab, j2h, j2t, j2b)

| kGat(kilNum, j1t, j1b)
| kGat(kilNum, j2t, j2b)
)
```

Matlab Code

1. Immune-Network-Based DNA Circuit.

```
x0 = [1 10 20 5 0 5];

afn = [0.0, 1.0, 0.0; 0.0, 0.0, 1.0; 1.0, 0.0, 0.0;
    0.3, 0.6, 0.1; 0.4, 0.4, 0.2; 0.1, 0.9, 0.0]

A1 = triu(rand(3));
A2 = transpose(A1);
A3 = A1 + A2;
A3p = rand(3);
A4 = A3p - diag(diag(A3p));
B = rand(3);

[t, x] = ode45(@(t, x) m1e(t, x, afn) ,[0 1.0], x0);
plot(t, x)
legend('A1', 'A2', 'A3', 'X', 'Y', 'Z')

function density = m1e (t, x, afn)

death = 0.0;

x1 = (afn(1,:) * [x(1); x(2); x(3)] - [x(1), x(2), x(3)] * afn(1:3,1) + afn(4,:)
 * [x(4); x(5); x(6)] - death) .* x(1);
x2 = (afn(2,:) * [x(1); x(2); x(3)] - [x(1), x(2), x(3)] * afn(1:3,2) + afn(5,:)
 * [x(4); x(5); x(6)] - death) .* x(2);
x3 = (afn(3,:) * [x(1); x(2); x(3)] - [x(1), x(2), x(3)] * afn(1:3,3) + afn(6,:)
 * [x(4); x(5); x(6)] - death) .* x(3);

x4 = -1 .* [x(1), x(2), x(3)] * afn(4:6,1) .* x(4);
x5 = -1 .* [x(1), x(2), x(3)] * afn(4:6,2) .* x(5);
```

```
x6 = -1 .* [x(1), x(2), x(3)] * afn(4:6,3) .* x(6);

density = [x1; x2; x3; x4; x5; x6];

end
```

2. Ant-System-Based DNA Circuit.

```
function ant_circuit

clc;
pA = 100;
pB = 120;
par = [pA pB];

[t, x] = ode45 (@rhs, [0 2], par);

c = 1;
sampling = floor((size(x, 1)/100));
output = zeros(sampling, 2);
for q=1:sampling:size(x, 1)
    output(c, 1) = x(q, 1);
    output(c, 2) = x(q, 2);
    c = c + 1;
end

plot (t, x);
xlabel('t'); ylabel('x');

function dxdt = rhs(t, x)
    k = 7.0;
    d = 100;
    ants = 10;
    a = 0; b = 0;
    for i=1:ants
        if rand > 0.8
            a = a + 1;
        else
            b = b + 1;
        end
    end

    pheroA = ((-1 .* k .* x(1)) + (d * a));
    pheroB = ((-1 .* k .* x(2)) + (d * b));

    dxdt = [pheroA; pheroB];
end

end
```

References

[1] David Yu Zhang and Georg Seelig. **Dynamic DNA nanotechnology using strand-displacement reactions**. *Nature Chemistry, 3, 103-113*, 2011. xi, 1, 2, 17, 48, 74, 78, 82

[2] Fei Zhang, Jeanette Nangreave, Yan Liu, and Hao Yan. **Structural DNA nanotechnology: state of the art and future perspective**. *J. Am. Chem. Soc., 136 (32), 1119811211*, 2014. xi, 6, 7, 11, 48, 74

[3] Nadrian C. Seeman. **Nucleic acid junctions and lattices**. *Journal of Theoretical Biology, 99(2), 237-247*, 1982. 1, 5

[4] Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman. **Design and self-assembly of two-dimensional DNA crystals**. *Nature, 394, 539-544*, 1998. 1, 5

[5] Bernard Yurke, Andrew J. Turberfield, Allen P. Mills Jr, Friedrich C. Simmel, and Jennifer L. Neumann. **A DNA-fuelled molecular machine made of DNA**. *Nature, 406, 605-608*, 2000. 1, 29

[6] Paul W. K. Rothemund, Nick Papadakis, and Erik Winfree. **Algorithmic self-assembly of DNA Sierpinski triangles**. *PLoS Biology 2(12), 2041-2053*, 2004. 1

[7] Paul W. K. Rothemund. **Folding DNA to create nanoscale shapes and patterns**. *Nature, 440, 297-302*, 2006. 1, 29, 48, 57

[8] Bryan Wei, Mingjie Dai, and Peng Yin. **Complex shapes self-assembled from single-stranded DNA tiles**. *Nature, 485, 623626*, 2012. 1

[9] Peng Yin, Harry M. T. Choi, Colby R. Calvert, and Niles A. Pierce. **Programming biomolecular self-assembly pathways**. *Nature, 451, 318-322*, 2008. 1, 30, 63, 74, 96

[10] David Soloveichik, Georg Seelig, and Erik Winfree. **DNA as universal substrate for chemical kinetics**. *PNAS, 107, 5393-5398*, 2010. 1, 20, 29, 30, 31, 49, 53, 54, 74, 75, 95, 96

[11] David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. **Computation with finite stochastic chemical reaction networks**. *Natural Computing, 7(4), 615-633*, 2008. 1, 97

[12] Georg Seelig, David Soloveichik, David Yu Zhang, and Erik Winfree. **Enzyme-free nucleic acid logic circuits**. *Science, 314(5805), 1585-1588*, 2006. 1, 8, 10, 29, 95

[13] Lulu Qian and Erik Winfree. **Scaling up digital circuit computation with DNA strand displacement cascades**. *Science, 332(6034), 1196-1201*, 2011. 1, 8, 17, 21, 29, 30, 48, 55, 78, 82, 95

[14] Ehsan Chiniforooshan, David Doty, Lila Kari, and Shinnosuke Seki. **Scalable, time-responsive, digital, energy-efficient molecular circuits using DNA strand displacement**. *DNA-16 LNCS, 6518, 25-36*, 2010. 1

[15] Ashish Goel and Mortega Ibrahimi. **Renewable, time-responsive DNA logic gates for scalable digital circuits**. *DNA-15 LNCS, 5877, 67-77*, 2009. 1

[16] Lulu Qian and Erik Winfree. **A simple DNA gate motif for synthesizing large-scale circuits**. *DNA-14 LNCS, 5347, 70-89, Springer: Heidelberg*, 2009. 1, 17, 30

[17] Lulu Qian, David Soloveichik, and Erik Winfree. **Efficient turing-universal computation with DNA polymers**. *DNA-15 LNCS, 5877, 67-77*, 2010. 1, 30, 41

[18] William B. Sherman and Nadrian C. Seeman. **A precisely controlled DNA biped walking device**. *Nano Letters, 4, 1203-1207*, 2004. 1, 30, 74

[19] Jong-Shik Shin and Niles A. Pierce. **A synthetic DNA walker for molecular transport**. *Journal of American Chemical Society, 126(35), 10834-10835*, 2004. 1, 30

[20] Peng Yin, Hao Yan, Xiaoju G. Daniell, Andrew J. Turberfield, and John H. Reif. **A unidirectional DNA walker that moves autonomously along a track**. *Angewandte Chemie Int. Ed., 43, 4906-4911*, 2004. 1, 30, 48, 74

[21] Suvir Venkataraman, Robert M. Dirks, Paul W. K. Rothemund, Erik Winfree, and Niles A. Pierce. **An autonomous polymerization motor powered by DNA hybridization**. *Nature Nanotechnology, 2, 490-494*, 2007. 1, 30, 48

[22] Tosan Omabegho, Ruojie Sha, and Nadrian C. Seeman. **A bipedal DNA brownian motor with coordinated legs**. *Science, 324(5923), 67-71*, 2009. 1, 8, 30

[23] Zhen-Gang Wang, Johann Elbaz, F. Remacle, R. D. Levine, and Itamar Willner. **All-DNA finite-state automata with finite memory**. *PNAS, 107(51) , 21996-22001*, 2010. 1

[24] Hongzhuo Gu, Jie Chao, Shou-Jun Xiao, and Nadrian C. Seeman. **A Proximity-based programmable DNA nanoscale assembly line**. *Nature, 465, 202-205*, 2010. 1, 74

[25] Satoshi Murata, Akihiko Konagaya, Satoshi Kobayashi, Hirohide Saito, and Masami Hagiya. **Molecular Robotics: A New Paradigm for Artifacts**. *New Generation Computing, 31 (1), 27-45*, 2013. 1, 48, 74, 95

[26] John H. Reif and Sudheer Sahu. **Autonomous programmable DNA nanorobotic devices using DNAzymes**. *Theoretical Computer Science, 410, 1428-1439*, 2009. 1, 48, 57, 95

[27] Rajiv Saini, Santosh Saini, and Sugandha Sharma. **Nanotechnology: the future medicine**. *Journal of Cutaneous and Aesthetic Surgery, 3(1), 32-33*, 2010. 5

[28] The Foresight Institute. **1995 feynman prize in nanotechnology awarded for pioneering synthesis of 3-D DNA objects**. *http://www.foresight.org/Updates/Update23/Update23.1.html (accessed 12/11/2015)*, 1995. 5

[29] Hao Yan, Sung Ha Park, Gleb Finkelstein, John H. Reif, and Thomas H. LaBean. **DNA-templated self-assembly of protein arrays and highly conductive nanowires**. *Science, 301(5641), 1882-1884*, 2003. 5

[30] Yu He, Ye Tian, Alexander E. Ribbe, and Chengde Mao. **Highly connected two-dimensional crystals of DNA six-point-stars**. *J. Am. Chem. Soc., 2006, 128(50), 1597815979*, 2006. 5

[31] Jianping Zheng, Jens J. Birktoft, Yi Chen, Tong Wang, Ruojie Sha, Pamela E. Constantinou, Stephan L. Ginell, Chengde Mao, and Nadrian C. Seeman. **From molecular to macroscopic via the rational design of a self-assembled 3D DNA crystal**. *Nature, 461, 74-77*, 2009. 5

[32] Thomas H. LaBean, Hao Yan, Jens Kopatsch, Furong Liu, Erik Winfree, John H. Reif, and Nadrian C. Seeman. **Construction, analysis, ligation, and self-assembly of DNA triple crossover complexes**. *J. Am. Chem. Soc., 2000, 122(9), 18481860*, 2000. 5

[33] Yu He, Yi Chen, Haipeng Liu, Alexander E. Ribbe, and Chengde Mao. **Self-assembly of hexagonal DNA two-dimensional (2D) arrays**. *J. Am. Chem. Soc., 2005, 127(35), 1220212203*, 2005. 5

[34] Jonathan Malo, James C. Mitchell, Catherine Venien-Bryan, J. Robin Harris, Holger Wille, David J. Sherratt, and Andrew J. Turberfield. **Engineering a 2D proteinDNA crystal**. *Angewandte Chemie Int. Ed., 44(20), 30573061*, 2005. 5

[35] Urmi Majumder, Abhijit Rangnekar, Kurt V. Gothelf, John H. Reif, and Thomas H. LaBean. **Design and construction of double-decker tile as a route to three-dimensional periodic assembly of DNA**. *J. Am. Chem. Soc., 2011, 133(11), 38433845*, 2011. 5

[36] Junghuei Chen and Nadrian C. Seeman. **Synthesis from DNA of a molecule with the connectivity of a cube**. *Nature, 350, 631-633*, 1991. 5

[37] R. P. Goodman, I. A. T. Schaap, C. F. Tardin, C. M. Erben, R. M. Berry, C. F. Schmidt, and A. J. Turberfield. **Rapid chiral assembly of rigid DNA building blocks for molecular nanofabrication**. *Science, 310(5754), 1661-1665*, 2005. 5

[38] Yu He, Tao Ye, Min Su, Chuan Zhang, Alexander E. Ribbe, Wen Jiang, and Chengde Mao. **Hierarchical self-assembly of DNA into symmetric supramolecular polyhedra**. *Nature, 452, 198-201*, 2008. 5

[39] Faisal A. Aldaye and Hanadi F. Sleiman. **Modular access to structurally switchable 3D discrete DNA assemblies**. *J. Am. Chem. Soc., 2007, 129(44), 1337613377*, 2007. 5

[40] Yuwen Zhang and Nadrian C. Seeman. **Construction of a DNA-truncated octahedron**. *J. Am. Chem. Soc., 1994, 116(5), 16611669*, 1994. 5

[41] William M. Shih, Joel D. Quispe, and Gerald F. Joyce. **A 1.7-kilobase single-stranded DNA that folds into a nanoscale octahedron**. *Nature, 427, 618-621*, 2004. 5

[42] Liping Feng, Sung Ha Park, John H. Reif, and Hao Yan. **A Two-State DNA Lattice Switched by DNA Nanoactuator**. *Angewandte Chemie Int. Ed., 115(36), 44784482*, 2003. 8

[43] Russell P. Goodman, Mike Heilemann, Soren Doose, Christoph M. Erben, Achillefs N. Kapanidis, and Andrew J. Turberfield. **Reconfigurable, braced, three-dimensional DNA nanostructures**. *Nature Nanotechnology, 3, 93-96*, 2008. 8

[44] Mathew M. Maye, Mudalige Thilak Kumara, Dmytro Nykypanchuk, William B. Sherman, and Oleg Gang. **Switching binary states of nanoparticle superlattices and dimer clusters by DNA strands**. *Nature Nanotechnology, 5, 116-120*, 2010. 8

[45] Dongran Han, Suchetan Pal, Yan Liu, and Hao Yan. **Folding and cutting DNA into reconfigurable topological nanostructures**. *Nature Nanotechnology, 5, 712717*, 2010. 8

[46] Fei Zhang, Jeanette Nangreave, Yan Liu, and Hao Yan. **Reconfigurable DNA origami to generate quasifractal patterns**. *Nano Letters, 12(6), 32903295*, 2012. 8

[47] Yangyang Yang, Masayuki Endo, Kumi Hidaka, and Hiroshi Sugiyama. **Photo-controllable DNA origami nanostructures assembling into predesigned multiorientational patterns**. *J. Am. Chem. Soc., 134(51), 2064520653*, 2012. 8

[48] Ye Tian and Chengde Mao. **Molecular gears: a pair of DNA circles continuously rolls against each other**. *J. Am. Chem. Soc., 126(37), 1141011411*, 2004. 8

[49] Arivazhagan Rajendran, Masayuki Endo, Kumi Hidaka, and Hiroshi Sugiyama. **Direct and real-Time observation of rotary movement of a DNA nanomechanical device**. *J. Am. Chem. Soc., 135(3), 11171123*, 2013. 8

[50] Kyle Lund, Anthony J. Manzo, Nadine Dabby, Nicole Michelotti, Alexander Johnson-Buck, Jeanette Nangreave, Steven Taylor, Renjun Pei, Milan N. Stojanovic, Nils G. Walter, Erik Winfree, and Hao Yan. **Molecular robots guided by prescriptive landscapes**. *Nature, 465, 206-210*, 2010. 8, 74

[51] David Yu Zhang, Andrew J. Turberfield, Bernard Yurke, and Erik Winfree. **Engineering entropy-driven reactions and networks catalyzed by DNA**. *Science, 318(5853), 1121-1125*, 2007. 8, 29, 48, 95

[52] Lulu Qian, Erik Winfree, and Jehoshua Bruck. **Neural network computation with DNA strand displacement cascades**. *Nature, 475, 368-372*, 2011. 8, 21, 22, 29, 30, 48, 65, 68, 78, 84

[53] Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. **Programmable chemical controllers made from DNA**. *Nature Nanotechnology, 8, 755-762*, 2013. 8, 22, 48, 55, 65, 74, 96

[54] Leonard M. Adleman. **Molecular computation of solutions to combinatorial problems**. *Science, 266, 1021-1024*, 1994. 8, 74

[55] Richard J. Lipton. **DNA solution of hard computational problems**. *Science, 268(5210), 542-545*, 1995. 9, 74

[56] Yaakov Benenson, Tamar Paz-Elizur, Rivka Adar, Ehud Keinan, Zvi Livneh, and Ehud Shapiro. **Programmable and autonomous computing machine made of biomolecules**. *Nature, 414(6862), 430-434*, 2001. 9

[57] Yaakov Benenson, Binyamin Gil, Uri Ben-Dor, Rivka Adar, and Ehud Shapiro. **An autonomous molecular computer for logical control of gene expression**. *Nature, 429(6990), 423-429*, 2004. 9

[58] Joseph N. Zadeh, Conrad D. Steenberg, Justin S. Bois, Brian R. Wolfe, Marshall B. Pierce, Asif R. Khan, Robert M. Dirks, and Niles A. Pierce. **NUPACK: Analysis and design of nucleic acid systems**. *Journal of Computational Chemistry, 32(1), 170-173*, 2010. 21, 78, 84

[59] Matthew R. Lakin, Simon Youssef, Filippo Polo, Stephen Emmott, and Andrew Phillips. **Visual DSD: a design and analysis tool for DNA strand displacement systems**. *Bioinformatics, 27(22), 3211-3213*, 2011. 22, 111

[60] Rizki Mardian, Kosuke Sekiyama, and Toshio Fukuda. **Multiple entities interaction model through DNA based petri net**. *Proceedings of the 5th International Conference on Automation, Robotics and Applications, 226-231*, 2011. 30, 31, 41, 97

[61] Rizki Mardian, Kosuke Sekiyama, and Toshio Fukuda. **DNA strand displacement for stochastic decision making based on immune's clonal selection algorithm**. *Proceedings of the 3rd International Conference on Natural Information Technologies*, 2012. 30

[62] J. Doyne Farmer; Norman H. Packard; Alan S. Perelson. **The immune system, adaptation, and machine learning**. *Physica 22D, 2, 187-204*, 1986. 30, 35

[63] Leandro N. D. Castro and Jon I. Timmis. **Artificial immune systems as a novel soft computing paradigm**. *Soft Computing, 7, 526-544. Springer: Heidelberg*, 2003. 30, 35

[64] Luca Cardelli. **Strand algebras for DNA computing**. *DNA-15 LNCS, 5877, 12-24. Springer: Heidelberg*, 2009. 31, 54, 56, 75, 97, 98

[65] Andrew Phillips and Luca Cardelli. **A programming language for composable DNA circuits**. *J. R. Soc. Interface*, 2009. 41, 65

[66] Richard A. Muscat, Jonathan Bath, and Andrew J. Turberfield. **A programmable molecular robot**. *Nano Letters, 11(3), 982-987*, 2011. 48, 74

[67] Shawn M. Douglas, Ido Bachelet, and George M. Church. **A Logic-gated nanorobot for targeted transport of molecular payloads**. *Science, 335 (6070), 831-834*, 2012. 48, 74

[68] Harish Chandran, Nikhil Gopalkrishnan, and John Reif. **DNA nanorobotics**. *Nanorobotics, 355-382*, 2013. 48, 63, 74

[69] Yonggang Ke, Luvena L. Ong, William M. Shih, and Peng Yin. **Three-dimensional structures self-assembled from DNA bricks**. *Science, 338(6111), 1177-1183*, 2012. 48

[70] Anthony J. Genot, Teruo Fujii, and Yannick Rondelez. **Scaling down DNA circuits with competitive neural networks**. *J R Soc Interface, 10(85)*, 2013. 48, 65

[71] Rizki Mardian, Kosuke Sekiyama, and Toshio Fukuda. **Approaching mathematical model of the immune network based DNA Strand Displacement system**. *Biosystems, 114(3), 245-252*, 2013. 49

[72] Marco Dorigo and Thomas Sttzle. **Ant Colony Optimization**. *MIT Press*, 2004. 49, 53

[73] Harish Chandran, Nikhil Gopalkhrishnan, Andrew Phillips, and John Reif. **Localized hybridization circuits**. *DNA 17, LNCS 6937, 64-83*, 2011. 50, 57

[74] Richard A. Muscat, Karin Strauss, Luis Ceze, and Georg Seelig. **DNA-based molecular architecture with spatially localized components**. *Proceedings of the 40th Annual International Symposium on Computer Architecture, 177-188*, 2013. 50

[75] Mario Teichmann, Enzo Kopperger, and Friedrich C. Simmel. **Robustness of localized DNA strand displacement cascades**. *ACS Nano, 8(8), 8487-8496*, 2014. 50

[76] E. Bonabeau, M. Dorigo, and G. Theraulaz. **Inspiration for optimization from social insect behavior**. *Nature, 406, 39-42*, 2000. 52, 53

[77] Luca Cardelli. **Two-domain DNA strand displacement**. *Mathematical Structures in Computer Science, 23(2), 247-271*, 2013. 55

[78] Eric Klavins. **Programmable self-assembly**. *IEEE Control Systems, 27(4), 43-56*, 2007. 74

[79] Y Kawauchi, M Inaba, and Toshio Fukuda. **A principle of distributed decision making of Cellular Robotic System (CEBOT)**. *Proceedings of IEEE International Conference on Autonomous Robots, 833-838*, 1993. 74

[80] Naoki Mitsumoto, Tadashi Hattori, T. Idogaki, and Toshio Fukuda. **Self-organizing micro robotic system: biologically inspired immune network architecture and micro autonomous robotic system**. *Proceedings of the 6th International Symposium on Micro Machine and Human Science, 261-270*, 1995. 74

[81] Naoki Mitsumoto, Toshio Fukuda, Fumihito Arai, and Tadashi Hattori. **Self-organizing multiple robotic system: a population control through biologically inspired immune network architecture**. *Proceedings of the IEEE International Conference on Robotics and Automation, 2, 1614-1619*, 1996. 74

[82] Christos G. Cassandras and Stephane Lafortune. **Introduction to discrete event systems**. *Kluwer Academic Publishers*, 1999. 74, 96

[83] Ronald C. Arkin. **Behavior-based robotics**. *The MIT Press*, 1998. 75

[84] Toshio Fukuda and Yoshio Kawauchi. **Cellular robotic system (CEBOT) as one of the realization of self-organizing intelligent universal manipulator**. *Proceedings of IEEE International Conference on Robotics and Automation*, 1990. 75

[85] Michael Wooldridge. **An introduction to multiagent systems**. *John Wiley Sons*, 2002. 75

[86] Paul Caplat, Madhur Anand, and Chris Bauch. **Symmetric competition causes population oscillations in an individual-based model of forest dynamics**. *Ecological Modeling, 211(3-4), 491-500*, 2008. 75

[87] Richard M. Crowder, Mark A. Robinson, Helen P. N. Hughes, and Yee-Wai Sim. **The development of an agent-based modeling framework for simulating engineering team work**. *IEEE Transactions on Systems, Man and Cybernetics, 42(6), 1425-1439*, 2012. 75

[88] Eric Bonabeau. **Agent-based modeling: methods and techniques for simulating human systems**. *PNAS, 99, 7280-7287*, 2002. 75

[89] Stuart J. Russell and Peter Norvig. **Artificial intelligence: a modern approach (2nd ed.)**. *New Jersey: Prentice Hall*, 2003. 75

[90] Hugo Costelha and Pedro Lima. **Modelling, analysis and execution of robotic tasks using petri nets**. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems, 1449-1454*, 2007. 75

[91] Hugo Costelha and Pedro Lima. **Robot task plan representation by Petri nets: modelling, identification, analysis and execution**. *Autonomous Robots, 33(4), 337-360*, 2012. 75

[92] V. A. Ziparo, L. Iocchi, D. Nardi, P. F. Palamara, and H. Costelha. **Petri net plans: a formal model for representation and execution of multi-robot plans**. *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, 2008. 75

[93] Tadao Murata. **Petri Nets: properties, analysis, and applications**. *Proceedings of the IEEE, 77, 541-580*, 1981. 75

[94] James L. Peterson. **Petri net theory and the modelling of systems**. *Prentice Hall, Inc.: Englewood Cliffs*, 1981. 75

[95] Matthew Cook, David Soloveichik, Erik Winfree, and Jehoshua Bruck. **Programmability of chemical reaction networks**. *Algorithmic Bioprocesses, 543-584*, 2009. 75

[96] David Yu Zhang. **Towards Domain-Based Sequence Design for DNA Strand Displacement Reactions**. *DNA 16 LNCS, 6518, 162-175*, 2011. 78

List of Publications

Title	Publication	Authors
I. Journals		
1. Ant Systems-Based DNA Circuits	BioNanoScience, Volume 5 (4), 2015, pp. 206-215	**Rizki Mardian** Kosuke Sekiyama
2. Model-Based Design and Control of Distributed DNA-based Systems by Petri Nets	Nano, Volume 11 (1), 2016	**Rizki Mardian** Kosuke Sekiyama
3. Approaching Mathematical Model of the Immune Network-Based DNA Strand Displacement System	Biosystems, Volume 114 (3), 2013, pp. 245-252	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda
4. DNA Strand Displacement for Stochastic Decision Making Based on Immunes Clonal Selection Algorithm	International Journal Information Theories and Applications, Volume 7 (1), 2013, pp. 34-45	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda

Title	Publication	Authors
II. International Conferences		
1. Probabilistic DNA-Based Gate Design for Stochastic Bio-molecular Computation	NIG International Symposium: Force, Information and Dynamics: X Factors Shaping Living Systems, January 9-11, Tokyo, Japan, 2016	**Rizki mardian** Kosuke Sekiyama
2. In-Vitro DNA-Based Finite State Machine	the 26th International Symposium on Micro-Nano Mechatronics and Human Science, Nagoya, Japan, 2015	**Rizki mardian** Kosuke Sekiyama
3. Ant Systems-Based DNA Circuits	the 21st International Conference on DNA Computing and Molecular Programming (DNA21), Boston, Massachussets, USA2015	**Rizki Mardian** Kosuke Sekiyama
4. DNA-Based Swarm Intelligence Inspired Computation	the 25th International Symposium on Micro-Nano Mechatronics and Human Science, Nagoya, Japan, 2014	**Rizki Mardian** Kosuke Sekiyama
5. Stochastic Computation for DNA-based Finite State Machine	the 20th International Conference on DNA Computing and Molecular Programming (DNA20), Kyoto, Japan, 2014	**Rizki Mardian** Kosuke Sekiyama

Title	Publication	Authors
6. DNA-Based Swarm Intelligence Inspired Computation	the 20th International Conference on DNA Computing and Molecular Programming (DNA20), Kyoto, Japan, 2014	**Rizki Mardian** Kosuke Sekiyama
7. Model of Reusable DNA Circuit that is Capable of Decision Making	the 11th Annual Conference on Foundations of Nanoscience: Self-Assembled Architectures and Devices, Snowbird, Utah, USA, 2014	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda
8. Approaching Mathematical Model of the Immune Network-Based DNA Strand Displacement System	the 19th International Conference on DNA Computing and Molecular Programming (DNA19), Tempe, Arizona, USA, 2013	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda
9. Stochastic Computation by Competing for Resource of DNA Population	the 7th International Workshop on Natural Computing, Tokyo, Japan, 2013	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda

Title	Publication	Authors
10. DNA Strand Displacement for Stochastic Decision Making Based on Immunes Clonal Selection Algorithm	the 3rd International Conference on Natural Information Technologies, Madrid, Spain, 2012	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda
11. Multiple Entities Interaction Model through DNA based Petri Net	the 5th International Conference on Automation, Robotics and Applications, Wellington, New Zealand, 2011	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda

Title	Publication	Authors
III.National Conference		
1. Petri Net for Designing Interaction with DNA Computing Principle	Annual Conference of the System Information, Tokyo, Japan, 2011	**Rizki Mardian** Kosuke Sekiyama Toshio Fukuda

Declaration

I herewith declare that I have produced this thesis without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This thesis has not previously been presented in identical or similar form to any other examination board.