Towards integrated molecular machines: structural, mechanical, and computational motifs

Thesis by
Namita Sarraf

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2023
Defended September 1st, 2022

© 2023

Namita Sarraf
ORCID: [0000-0001-8692-7429]

All rights reserved

ACKNOWLEDGEMENTS

It was a serendipitous moment of my early graduate school career that I got randomly added to Lulu Qian's schedule during interview weekend - though it took me a detour of a year or two to end up finding my way first to Lulu's class, then to her lab. I have been exceedingly fortunate to work with and learn from Lulu. There are so many things I admire about her: how pure her motivation for doing great science is; her dedication to conveying her science and vision in an artistic and aesthetic way; her genuine support and care for her students and postdocs. Her rigor and attention to detail are both things I deeply admire and will continue to try to emulate. In her hands, science is beautiful. I could not have asked for a better mentor to guide me through the professional and personal ups and downs of graduate school and help me develop my own mentorship skills as well. Thank you, Lulu, for patiently guiding me along this path and giving me an academic home that I am so grateful and proud to have been a part of. (And for introducing me to the best Chinese restaurants in LA!)

I would also like to thank my first advisor in graduate school, Julia Greer, for cultivating such a welcoming and loving scientific community that I was lucky to join and for giving me your staunch support when I started to find my interests wandering. Your encouragement to pursue my passions gave me the courage to change trajectories, as sad as I was to leave your lab.

Thank you to Paul Rothemund, who lent me an ear and crucial advice at an inflection point in my graduate career, when I was hovering between two labs and unsure of what direction to take. Working with you and Ashwin for the short time that I did was an invaluable experience.

Thanks to my other committee members, Richard Murray and Mikhail Shapiro, who have each provided me with helpful feedback and advice throughout graduate school, but particularly near the beginning - Richard when rotating in his lab, and Mikhail as my first-year advisor.

I have been fortunate to work with a great many other incredible scientists during my time at Caltech. Nathan Schoepp and Daryl Yee, who were my first mentors in the Ismagilov and Greer labs, respectively, and both of whom turned into trusted friends. In the Qian lab, Grigory Tikhomirov, Philip Petersen, and Anu Thubagere, whose elegant work I have ridden the coattails of. A particular thanks to Greg for passing on your AFM wisdom to me. And to Philip, it has been a pleasure getting the chance to work with you in the last year. Thanks to Kevin Cherry for sharing his wealth of knowledge about the science and the people in the field, for being my go-to person to bounce experimental ideas (or any ideas, really) off of, and for teaching me how to change the oil in my car. I am not sure I would have survived graduate school without you. And thanks to Kellen Rodriguez, an undergraduate working with me who has proven himself a capable and insightful researcher and who is now headed off to program some molecules in graduate school himself. I will always be excited to learn what exotic animal or tree you acquire next. Thanks to the other Qian Lab and DNA Lab (Winfree/Rothemund) members for cultivating a spirit of scientific community and a supportive environment in which to learn and grow.

A special thanks to the ladies of Keck who have taken care of me all these years - Miki Yun, who has always made sure the Echo was running smoothly, and Christy Jenstad, whose door and arms are always open.

To Acacia Hori and Jessica Griffiths, with whom we started Women in BBE - you both continue to inspire me with your dedication to community service. I am grateful to have found allies in this cause, as well as great friends.

On a personal note - thanks to my first roommate in the Cats, Lucía DeRose, who has been there through all the ups and downs. To the Wam girls, who felt like my first real identity here. To Alison Koontz and Christian Kuntz, who are my chosen family. To Annalise Sundberg and Paulomi Bhattacharya, my home base during most of the pandemic. To Siobhán MacArdle, Elise Tookmanian, Luke Metzler, Katrina Hui, Katherine Rinaldi – there are too many people to name who have loved, laughed, and commiserated with me, and been sucked into my constant pursuit of the best food in the city.

Finally, the biggest thank you goes to my family. To my dad, the original Dr. Sarraf, who has always supported, loved, and challenged me, and picked me up when I've fallen. To my mom, my best friend, confidante, and biggest cheerleader. To my brother, Nimish, my polar opposite, the biggest goof, without whom I cannot imagine going through this life. To my extended family in LA and SF, who have made me feel at home on a strange coast. And to my many aunts and uncles back home. Thank you for always believing in me and always being proud of me. My large, hectic, loving, silly community is the essence of what keeps me going.

ABSTRACT

The programmability of DNA has made it well-suited for building molecular machines, performing nanoscale self-assembly, and computing via biochemical circuits. In the last few decades, great strides have been made in characterizing the interactions between DNA molecules such that they can be predicted and engineered. The development of frameworks for those interactions has enabled the construction of more complex molecular systems that can execute specified programs. Such programs have included mechanical tasks, like walking and sorting cargo; assembly and reconfiguration of 2D and 3D shapes; and computation, like Boolean logic and pattern recognition.

However, the continuing development of more complex molecular programs relies upon expanding the modules available for molecular systems to use to execute them. Expanded functionality of mechanical, structural, and computation modules are required in order to build compound systems that can interact with the physical world, reconfigure, and analyze input signals in a variety of interesting ways. In this dissertation, we will discuss our contributions to this effort, which include exploring a motif for molecular robotic behavior, characterizing tile-tile interactions, and developing new capabilities for bimolecular circuits.

Within the framework of a maze-solving molecular robot, we aim to implement walking behavior on DNA origami that introduces a surface modification via a four-way strand displacement reaction. Surprisingly, our experiments suggest that the walking behavior is at least two orders of magnitude slower than expected. To understand why, we quantitatively explore to what extent the speed and completion level of the robot can be modulated by design considerations such as toehold lengths, track redundancy, and strand purity. Another factor affecting the reaction rate is the number of tethering points, and we demonstrate an order of magnitude speed up in the four-way strand displacement reaction when we remove one tethering point. The characterization of a surface-modifying four-way strand displacement reaction is a useful tool for the continued development of molecular robots with more complex functionality.

Free-floating DNA origami tiles, called invaders here, can swap out DNA origami tiles within larger assemblies via a technique called tile displacement, which has previously been demonstrated using single tile and dimer invaders with 4- and 9-tile arrays. We introduce initial structures and invading assemblies with more complex shapes. We explore the robustness of this reaction by testing a variety of edge configurations and comparing their reaction rates. We demonstrate tunable growth of one of the invaders, which can grow into polymers of arbitrary length or close into 3D structures. By a tile displacement reaction, we reconfigure the 3D structures into 2D. The invaders with complex shapes are able to reconfigure the original tile assembly at rates comparable to simpler tile displacement reactions, and two reconfiguration events can take place sequentially or simultaneously.

Finally, we build two new modules for use with biochemical circuits. The first, a loser-take-all circuit, yields binary outputs indicating which analog signal is the smallest among all inputs. We implement a signal reversal function that converts the smallest input to the largest output, which can then be composed with a previously developed winner-take-all function to achieve loser-take-all. By making concentration adjustments, we can mitigate biases in the circuit that are a result of sequence-dependent different in reaction rates. We experimentally demonstrate a three-input loser-take-all circuit with nine input combinations. With further development, this circuit could be used to implement the activation function in neural networks that perform pattern classification according to which memory an input pattern is least similar to.

The second circuit processes information using temporary memory. We design and implement a circuit that outputs distinct logic decisions based on relative timing information of a pair of inputs and their logic values. We show that we can mitigate crosstalk in the circuit by utilizing mismatches and adjusting toehold lengths. The circuit is able to display clear ON-OFF separation at time intervals as short as one minute between the two inputs arriving.

PUBLISHED CONTENT AND CONTRIBUTIONS

Lapteva, Anna P.*, Namita Sarraf*, and Lulu Qian (2022). “DNA Strand Displacement Temporal Logic Circuits”. In: *J. Am. Chem. Soc.* 144.27, pp. 12443–12449. DOI: `10.1021/jacs.2c04325`.
* Equal contribution
A.P.L. and L.Q. initiated the project; A.P.L. wrote the first draft of the manuscript; N.S. proposed the experiments on balancing reaction rates with more even spread of cytosines; N.S. performed most of the experiments; A.P.L. performed the experiments on varying time intervals between inputs; all authors designed the experiments, analyzed the data, and edited the manuscript; L.Q. guided the project.

Sarraf, Namita, Kellen R. Rodriguez, and Lulu Qian (2022). “Tile-displacement-based shape reconfiguration in DNA origami tile assemblies”. In: *in preparation*.

Rodriguez, Kellen R.*, Namita Sarraf*, and Lulu Qian (2021). “A Loser-Take-All DNA Circuit”. In: *ACS Synth. Biol.* 10.11, pp. 2878–2885. DOI: `10.1021/acssynbio.1c00318`.
* Equal contribution
K.R.R. initiated the project and wrote the first draft of the manuscript; N.S. performed the experiments; all authors designed the experiments, analyzed the data, and edited the manuscript; L.Q. guided the project.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

NOMENCLATURE

AFM. Atomic force microscope, which is a physical imaging instrument that is used to visualize DNA origami.

Branch migration. The random walk displacement of one bound strand by another. It is the second step of strand displacement, after toehold binding.

CRN. Chemical reaction network. A set of reactions that describe a chemical system.

DNA origami. A concept in which a very long strand of DNA, called a scaffold, is pulled into a specific shape by many smaller strands, called staples, that are complementary to the scaffold strand in two non-adjacent locations.

Enthalpy. A thermodynamic measure of energy trapped in bonds. A system with more base pairs bound vs. free nucleotides has less enthalpy. A reaction is more likely to be favored if it decreases enthalpy, e.g. results in more base pairs bound.

Entropy. A thermodynamic measure of randomness. More entropy can mean more degrees of freedom (e.g. a single DNA strand has more entropy than a duplex because it is more flexible) or a larger number of complexes. A reaction is more likely to be favored if it increases entropy.

Four-way strand displacement. Two double-stranded complexes hybridize to form a Holliday junction and exchange their strands. See Section 1.6 for more detail.

Hairpin. A single strand of DNA that is complementary to itself. When it binds to itself, it forms a hairpin-like loop at one end.

PAGE gel purification. Polyacrylamide gel purification is a technique by which DNA molecules are separated by size using an electrical gradient to push negatively-charged DNA through a chemically crosslinked matrix.

Strand displacement. A mechanism by which one strand replaces another in a DNA complex.

Surface CRN. A chemical reaction network that is tethered to a surface such that it is spatially organized.

Three-letter code. The three-letter code minimizes secondary structure within a single strand. Regular domains (e.g. leg) only use A's, T's, and C's, while star domains (e.g. leg*) use A's, T's, and G's.

Toehold. In strand displacement, a short single-stranded sequence at the end of a longer strand that an invading strand can bind to in order to initiate strand displacement. In tile displacement, a series of 2-nucleotide truncations on edge staples that allow an invading tile to bind to the exposed scaffold loop.

Chapter 1

INTRODUCTION

1.1 Programming molecules

It is profound that DNA, a molecule with such a simple code – made up of A's, T's, C's, and G's – contains the blueprint for anywhere from tens of thousands to billions of proteins distinct in both form and function that are essential for human operation. Even more impressive is that it can give rise to machinery as complicated as the human brain. It is a beautiful notion that such complexity can arise from such simplicity – like great art arises from simple colors on a canvas.

There are many examples of computation in biology at the cellular and organism-wide level. Cells are continuously computing ratios of proteins, like NADH and NAD^+, to regulate metabolism, which requires that they do a form of division. John Hopfield proposed that cells do kinetic proofreading to maintain high fidelity in DNA replication, akin to error-correcting code (Hopfield, 1974), which was later corroborated with evidence. Ants and honey bees use quorum sensing to efficiently drive group behavior (Franks et al., 2015), which has inspired methods for guiding the behavior of autonomous swarm robots. And, of course, the way that neurons process information is the inspiration for artificial neural networks in modern computer science, though the mechanisms are quite different.

In those first encountering the superimposition of biology and computer science, a natural comparison arises between DNA sequences, which is made up of nucleotides and directs a cell's functioning, and machine code, which is made up of 0's and 1's and directs the computer's central processing unit. The major difference is that DNA, besides being an information-carrying molecule, also interacts directly with matter and has the capability of influencing its physical world, whereas electronic computers rely on layers of abstraction that separate their core functionality from their physical substrate. In other words, the physical substrate provides the infrastructure needed to perform the computation but cannot be fundamentally altered by it – nor would it be useful if it were[1]. Said even more simply: in computers, software and hardware are separate; in biology, they are the same (Bray, 2011).

[1]This is particularly convincing when we consider that there are many types of physical substrates that can support logic, including water waves, Legos, the original vacuum tubes, and even soldier crabs.

This difference becomes clearer when we consider the types of tasks that biology can perform that electronic computers cannot. Cells execute complex programs to grow, divide, and heal, none of which have been achievable yet by a laptop or desktop. So along with the notion that biology already performs complex computations, and borrowing the paradigm from computer science that we can formalize a set of instructions to produce a desired end result, the question arises – can we exploit the machinery of biology to design rational systems? And can we program those systems to perform functions that are currently impossible with the tools we have?

While such a question might be inspired by natural processes, its execution will necessarily look quite different. We can take advantage of the programmability of DNA while building a system that looks nothing like what you might find in a cell. Our goal is not to discover how complex of a system nature can build. Rather, it is to discover what kinds of systems we can build with the understanding we have of nature's building materials. How complex of functionality can a system of molecules as simple as DNA achieve?

1.2 The interplay of structure and function

The dream of building molecular computers that compute using chemistry, rather than electronics, has been rooted in the notion that the chemistry inside living beings is already able to carry about complex computation, such as the division and proofreading discussed in the previous section. Long before any experimental demonstration of a molecular computer, Charles Bennett observed that biopolymers like DNA and RNA can behave in a remarkably analogous way to automata, such as a Turing machine (Bennett, 1973). He theorized that it would be possible to build such devices out of biomolecules given enzymes with the right capabilities (Bennett, 1982). The first experimental demonstration of DNA computing being used to solve a computational problem was by Len Adleman in 1994, in which he used a system of synthetic DNA and enzymes to solve the NP-complete Hamiltonian path problem (Adleman, 1994). Soon after, a similar strategy was used to solve another NP-complete problem, the satisfiability problem (SAT) (Q. Liu et al., 2000).

The first person to realize that DNA could also be exploited structurally, and designed to self-assemble into useful junctions and shapes, was Ned Seeman, who was a young professor at SUNY Albany at the time. It dawned on him over a beer at the campus pub that there was a similarity between six-arm DNA branched junctions and M. C. Escher's periodic array of flying fish in "Depth," shown in Figure 1.1. This inspired him to consider that there may be a way to direct the structure and crystallization of DNA. He had the vision for how DNA could be used for massively parallel self-assembly at the nanoscale, as well as structural transitions that would be key to building nanomechanical devices (Seeman, 1998). Inspired by this paradigm and exploiting the well-characterized Watson-Crick complementarity of DNA, self-assembly methods were developed that first created crystals (Winfree, 1998), then specific shapes, called DNA origami (Rothemund, 2006).

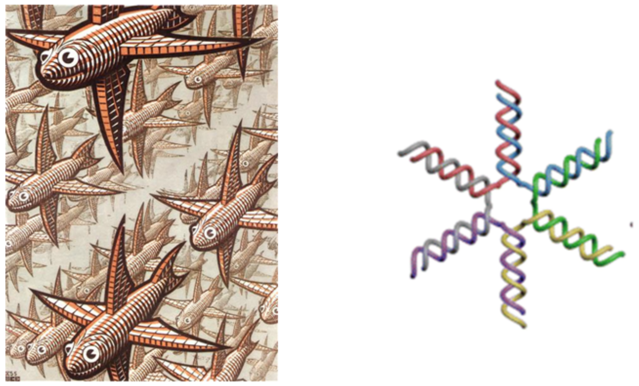


Figure 1.1: Comparison of M. C. Escher's art with DNA junctions. Left, "Depth" by M. C. Escher. Right, six-armed DNA branched junctions, adapted from Seeman and Gang, 2017. Ned Seeman was inspired by the similarities he noticed between the flying fish in "Depth" and the branched DNA junctions to develop a rational approach to assembling DNA structures.

Algorithmic self-assembly is perhaps the earliest example that emulates the intimate relationship between structure and computation. This trajectory of the field was born when Erik Winfree made the connection between the edges of Wang tiles, whose self-assembly acts like a Turing machine (H. Wang, 1963) and the sticky edges of DNA (Winfree, 1998). Early on it was used to execute, for example, basic logical operations (Mao et al., 2000), while eventually it was shown to be able to implement arbitrary cellular automata (Rothemund, Papadakis, and Winfree, 2004). However, these advances were dependent on simultaneous advances in our understanding of how to build DNA structures. The use of the antiparallel double crossover as a way to provide structural rigidity in self-assembled tiles (Fu and Seeman, 1993; X. Li et al., 1996) led to the use of double-crossover (DX) tiles to crystallize self-assembled DNA structures (Winfree, F. Liu, et al., 1998). DX tiles and triple-crossover (TX) tiles have become the basis for demonstrating universal computation using tile assembly models (cite).

Like a DNA junction, the focus of specific lines of inquiry have branched in various directions as the field has matured. But in many of those directions, the lines between structure and function have become increasingly blurred. Algorithmic behaviors have been implemented on top of tiles, for example, that require the development of new tile structures, as in a cargo-sorting molecular robot that is built on a double-layer DNA origami tile that must be more rigid than the more traditionally used single layer tile in order for the robot to function as desired (Thubagere et al., 2017). Moreover, DNA origami tiles have been used to reconfigure other tile structures according to some set of instructions, as in a tic-toe-game played by using tile displacement (Petersen, Tikhomirov, and Qian, 2018). It is a poetic extension of the idea that in biology, software and hardware are the same; in many synthetic biomolecular systems, they are equally intertwined.

1.3 Molecular robots and tile displacement

Molecular robots are one such system. They are inspired by the behavior of naturally-occurring molecular machines, like kinesin, which shuttles cargo around cells via microtubule walkways. They are designed to perform a mechanical task, and rely upon surface-based reactions in order to do so. The first devices that demonstrated the kind of movement that would be necessary to build robots were tweezers that could be opened and closed (Yurke, Turberfield, et al., 2000), while early demonstrations of walkers moved along a track via the addition of fuel strands that powered each step (Shin and Pierce, 2004), or via ATP hydrolysis (Yin et al., 2004). Following robots were developed to be some combination of fast, autonomous, bidirectional, processive, and reusable (Omabegho, Sha, and Seeman, 2009; Lund et al., 2010; Wickham et al., 2012; J. Li et al., 2018). As the mechanisms for building DNA robots that could walk along a track became better understood, the outlook turned towards adding in other functionality as well. One robot was designed that could pick up cargo off of an assembly line based on whether it was in an "on" or "off" state (Gu et al., 2010). Another could transport cargo along a track according to instructions from fuel molecules (Muscat, Bath, and Turberfield, 2011). Finally, the cargo-sorting robot was able to autonomously pick up randomly distributed cargo on a DNA origami surface and sort it into bins (Thubagere et al., 2017).

While the types of tasks that molecular robots can perform have begun to increase in complexity as walking behavior is combined with skills like picking up and sorting cargo, the library of functional modules that we can draw upon to achieve additional functionality is quite limited. The new abilities of the cargo-sorting robot were designed in such a way that they could be easily composed together with previous and future skills. Similarly, we hope to expand the toolbox for molecular robots by designing modular skills that can be pulled off the shelf and easily composed together to build robots with more interesting functionality. To that end, in Chapter 2 we will discuss the exploration of a new skill, maze-solving, and its physical implementation, adding a track modification marking the solved path.

The other case mentioned above is tile displacement, which is a reaction involving the interaction of edges of DNA origami tiles that alters the structures of tile arrays. Because of their programmability and addressibility, in that molecules can be placed at specific locations on them, DNA origami tiles have been used to fabricate nanoscale devices (Knudsen et al., 2015), organize molecular circuits (Chatterjee et al., 2017), and provide the track for molecular robots (Thubagere et al., 2017). They can also be used to create random infinite arrays with global properties (Tikhomirov, Petersen, and Qian, 2017b) and micron-scale 2D structures via hierarchical assembly (Tikhomirov, Petersen, and Qian, 2017a). And, similar to the interactions of strands in strand displacement reactions, it has been shown that tiles can interact via their edges to perform swap reactions, called tile displacement reactions (Petersen, Tikhomirov, and Qian, 2018).

So far, the tile displacement mechanism has been demonstrated on 4 and 9-tile arrays with single tile or dimer invaders, which opens up the possibilities of being able to perform structural reconfiguration in an already formed array. However, we still do not know how certain characteristics of these systems, such as the edge identities of the tiles, affect the reaction speed and completion level. In Chapter 3, we establish principles for designing such systems and expand the scope of tile displacement to origami assemblies with more complex shapes.

1.4 Modulating the behavior of structural systems

It has been previously considered that one advantage of molecular computers is their ability to interface with their physical world. For example, the input to a molecular program could be another molecule, such as DNA or RNA, which informs when the computation should happen as well as what the output should be. However, with molecular robots and tile displacement systems, behavior is encoded into the local environment of the structure, so the machinery that they have to analyze signals from the more global environment in which they are located is limited. The assembly line robot responds to environmental signals that dictate whether the state of each two-state stop along the line is "OFF" or "ON." The cargo-sorting robot responds to an environmental signal to execute its program. Strands that help other robots choose their paths could also be seen as environmental signals. And it has been shown that theoretically, DNA robot circuits can be programmed to process more complex environmental signals (Dannenberg et al., 2015). However, experimental demonstrations of the information-processing capabilities of molecular robots and tile displacement systems thus far have been limited to relatively simple functions.

One could imagine interfacing a decision-making circuit with a structural system such that it can be modulated by environmental signals that have been analyzed via more complex information-processing functions. There is a class of reaction systems, well-mixed strand displacement cascades, that are particularly well-suited to recognize input signals, perform a computation or make a decision, and produce an output. Not only can they interface with a variety of molecules, including DNA, RNA, and proteins, but they are also often dependent on simple, two or at most three-stranded molecules, which lends itself to scalability. If this type of system could be integrated with molecular robots and tile displacement systems, it could provide more sophisticated control over their behavior.

One framework with which to design such reaction systems is via the seesaw DNA gate motif, which can be used to demonstrate digital logic circuits and neural network computation (Qian, Winfree, and Bruck, 2011; Qian and Winfree, 2011). Other strand displacement-based reaction networks that leverage mechanisms like cooperative hybridization have proven effective as well – for example, to build a classifier for disease states based on detected gene expression (Lopez, R. Wang, and Seelig, 2018). To add to the library of modules that we can choose from in designing information-processing circuits, we explored two kinds of circuits that extend the capabilities of strand-displacement cascades: first, a loser-take-all circuit, and second, a temporal logic circuit.

A winner-take-all neural network is a type of information-processing circuit that is capable of classifying patterns based on its memories, which is a useful tool for determining whether environmental signals match a specified profile. It is well-suited for strand displacement networks because it does not require dual-rail representation to represent negative weights. It has been demonstrated that a molecular neural network can be built using the seesaw motif that recognizes and classifies 100-bit patterns (Cherry and Qian, 2018). However, one limitation of the previously implemented winner-take-all network is that its weights are averaged training patterns, so if two memories are very similar, it may not be possible to distinguish between them. When it is not possible to classify a pattern because of the similarity between two or more classes, it may be useful to instead determine which memory the input pattern is least similar to. We call this logic loser-take-all, as it is the inverse of the winner-take-all function. In Chapter 4, we implement a loser-take-all circuit.

Another useful indication from the environment is the relative timing of different signals. This kind of temporal information is used widely in biology, from echolocation to regulating gene expression. Previously demonstrated DNA circuits that process timing information have used two different strategies: cross-inhibition and temporal memory. In cross-inhibition, when there are two inputs, one input signal inhibits the other (C. Liu et al., 2020). However, this method is dependent only on which signal arrives first, and would, for example, produce the same output if a second input arrived later or never arrived at all. In contrast, in temporal memory, the output of the circuit depends on the relative arrival time of all inputs, so will provide a distinct output if one input were to arrive later or never arrive. Temporal memory has been demonstrated in a DNA circuit that uses a polymerase to encode the information into DNA strands (Kishi et al., 2018). Our aim was to construct a temporal memory-based circuit that uses only DNA molecules, and we show this implementation in Chapter 5.

Beyond developing new strand displacement modules that can be used to control molecular robots and tile displacement systems, the next step is to build the mechanism for interfacing the two types of systems. While we have begun to explore this alongside other researchers, it falls outside of the scope of this thesis.

1.5 Summary of this thesis

In this thesis we will discuss several distinct, but related, systems, and imagine how they might fit together to create structural devices that are responsive to their environments. Here we will summarize our main contributions.

Chapter 2 shows the algorithmic design and experimental exploration of a maze-solving DNA robot that adds a surface modification to the track as it walks. We first explored a previously proposed design that uses three-way strand displacement, then designed a new robot that uses four-way strand displacement. We characterized reaction rates for irreversible and reversible track-modifying four-way strand displacement reactions on a surface. Finally, we explored to what extent design considerations such as toehold lengths, track redundancy, and strand purity can be leveraged to improve the speed and completion level of the robot.

Chapter 3 demonstrates reconfiguration of a tile assembly when both the initial assembly and invading assemblies have a complex shape. We tested different edge identities, toehold, and branch migration configurations to determine reaction rates, and establish design principles based on these findings. By tuning the tile ratios of an assembly that can grow arbitrarily long, we showed that we can tune the length distribution of the polymer. By adding tile displacement toeholds on the edge of a tile that is part of a 3D structure, we showed that we can use tile displacement to reconfigure a 3D structure to 2D. Finally, we demonstrated that given a tile assembly with two tile displacement toeholds, we can simultaneously reconfigure it with the same efficiency as if we were reconfiguring it by just one toehold.

In Chapter 4, we implement a molecular loser-take-all circuit that yields binary outputs indicating the smallest analog signal among all inputs. Given n inputs, each output corresponds to the average of $n - 1$ inputs. As a result, the largest output signal from the signal reversal layer will come from the averages of all but the smallest input, and thus correspond to the smallest input. We showed that we can balance the reaction rates within each layer of the circuit by adjusting species concentrations, which is essential for correct computation. Finally, we successfully implemented a three-input circuit with nine input combinations.

Chapter 5 shows that we can build a DNA strand displacement circuit that processes information using temporal memory. We designed a circuit that can output distinct logic decisions based on relative timing information for a pair of inputs and their logic values. The output is produced as the result of a cooperative hybridization between one input and the memory of the other input. There was some crosstalk between the reaction pathways for the two inputs, so we used a mismatch method to mitigate it. More detailed models separating toehold binding and dissociation from branch migration indicated that we could suppress crosstalk further by shortening the toehold on one side of the cooperative hybridization gate, which did increase the separation between the ON and OFF states of the circuit. Finally, we showed that the circuit displayed a clear ON-OFF separation even when the time interval between the two inputs arriving was shortened to 1 min.

1.6 Review of essential concepts

In this section, we review several concepts that are integral to the systems that will be discussed in subsequent chapters.

Three-way toehold-mediated strand displacement

Toehold-mediated strand displacement is a key tool that is used to introduce structural changes and kinetics into DNA-based systems. In strand displacement, a single strand or complex hybridizes to a single or both strands in a duplex, replacing the strand in the duplex or exchanging strands. This mechanism has been studied in the context of biological processes since the 1970's (Redding et al., 1977; Panyutin and Hsieh, 1994). However, it was not until toeholds were introduced by Yurke *et al.* as a way to control strand displacement kinetics and reuse molecules that strand displacement became very useful in DNA nanotechnology (Yurke, Turberfield, et al., 2000).

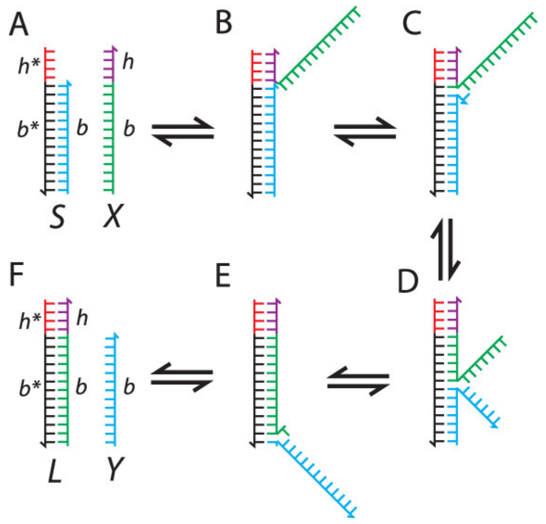


Figure 1.2: Toehold-mediated strand displacement via three-way branch migration. (A) The toehold consists of the single-stranded overhang $h*$ on the duplex S. The invader strand X displaces the incumbent strand b, also denoted as Y to form the duplex L in (F). (B-E) Intermediate steps in the branch migration process. This figure is adapted from (Simmel, Yurke, and Singh, 2019).

The mechanism of toehold-mediated three-way strand displacement is illustrated in Figure 1.2. The short toehold domain, h, of a single strand, X, binds to an exposed single-stranded region, $h*$, at the end of the longer strand in the duplex S. The domain b on X then competes with domain b on S to be bound to $b*$ via a random walk displacement. Finally, the original cover strand b is fully displaced and floats away. This process is illustrated in Figure 1.2.

By varying the length and sequence of a toehold, the kinetics of strand displacement can be tuned over six orders of magnitude (David Y. Zhang and Winfree, 2009).

Four-way toehold-mediated strand displacement

In contrast to three-way strand displacement, in which a single strand invades a complex, in four-way strand displacement two complexes exchange strands with each other.

The use of four-way branch migration in homologous genetic recombination was first proposed by Robin Holliday in 1964 (Holliday, 1964). The structure subsequently named a Holliday junction forms, which is a branched nucleic acid structure made up of four strands. There are three conformations that this junction can take: parallel stacked-X, open-X, and antiparallel stacked-X, as shown in Figure 1.3. In high salt conditions, the junction adopts a parallel or antiparallel stacked-X form, in which the arms coaxially stack into two parallel double helices. In the open-X form, the four duplex arms, experiencing repulsion from each other's negatively charged phosphate backbones, are extended at 90° angles from each other (Hays, Watson, and Ho, 2003). In the antiparallel form, the cross-over strands form a U-turn, which prevents the junction from migrating along the DNA strands.

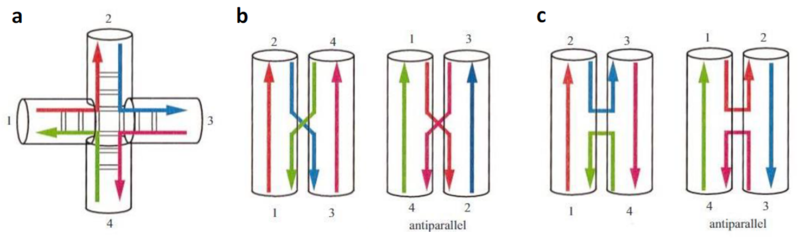


Figure 1.3: Structural forms of DNA Holliday junctions. (a) The extended open-X form. (b) The parallel stacked-X form, showing the two possible pairwise coaxial stacking configurations. (c) The antiparallel stacked-X form, also showing the two possible stacking configurations. This figure is adapted from (Lilley, 2000).

Figure 1.4a shows the mechanism proposed by Holliday by which homologs trade strands[2], while b and c show more detailed diagrams of the junction configurations. He originally hypothesized about the parallel stacked-X form of the junction, in which the swapping strands cross each other. In the four-way strand displacement reactions that we engineer, we exploit a similar mechanism as Holliday proposed, though without the nicking step. We use a standard 12.5mM Mg^{2+} concentration, which is high enough to shield the backbone charge and allow close packing of

[2]Interestingly, because of Holliday's discovery, the first studies of DNA branch migration were done on four-way branch migration reactions, though at present three-way branch migration is much more widely used in DNA nanotechnology.

helices as in the antiparallel stacked-X junction. We will discuss the junction configuration in more detail in Chapter 3.

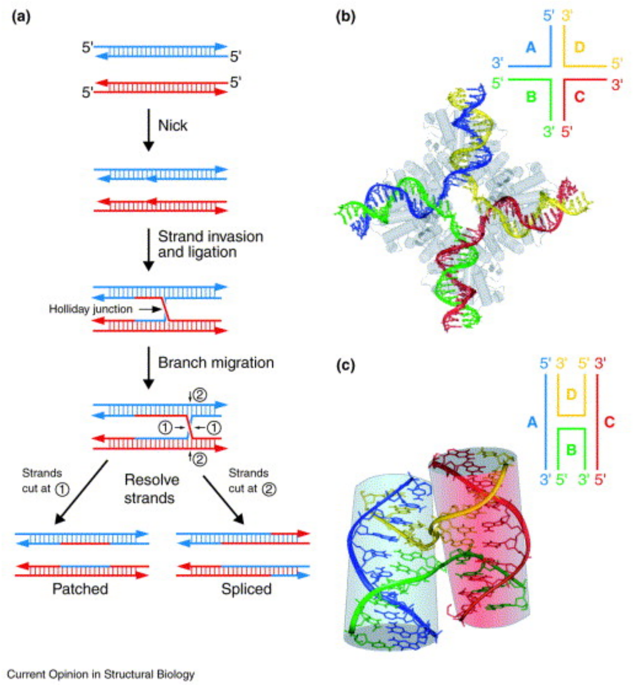


Figure 1.4: Recombination and the Holliday junction. (a) The model for homologous recombination proposed by Holliday. (b) The open-X junction. (c) The antiparallel junction. This figure is adapted from (Ho and Eichman, 2001).

The mechanism of four-way strand displacement is shown in Figure 1.5. Two duplexes with matching toeholds, here *b* and *c*, bind to form a Holliday junction. As the four-way branch migration proceeds, the branch point moves until eventually two new duplexes are created. In this case, the four-way strand displacement results in a greater number of paired bases than the starting complexes, though as in three-way strand displacement that is not always the case.

DNA origami

DNA origami is a technique that was invented by Paul Rothemund in 2006. A long scaffold strand, commonly M13mp18, a viral genome that is 7249 nucleotides long, is pulled into a designed 2D shape by shorter staple strands that are complementary to the scaffold (Rothemund, 2006). This is illustrated in Figure 1.6.

Using this technique, it is possible to form any arbitrary 2D shape. The technique has also been extended to 3D shapes, but that is outside the scope of this thesis. Here,

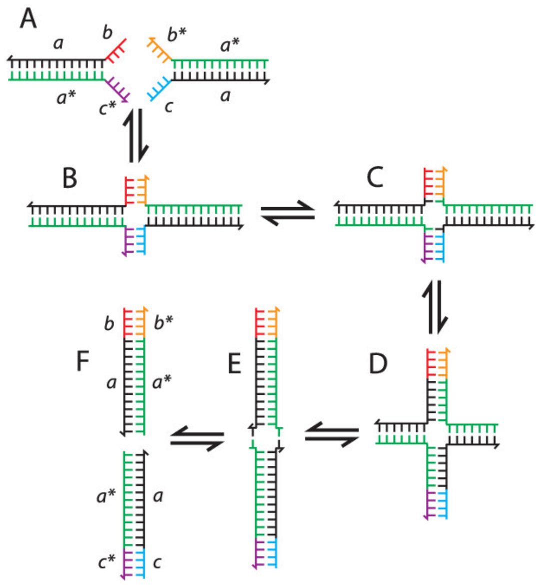


Figure 1.5: Toehold-mediated strand displacement via four-way branch migration. (A) The toeholds consist of the single-stranded overhangs b and $c*$ on the duplex a. (B) The two complexes bind at the toeholds to form a Holliday junction. (C-E) show the random walk junction migration as the branch point moves through four-way branch migration. The result is two new duplexes (F) that have gained base pairs. This figure is adapted from (Simmel, Yurke, and Singh, 2019).

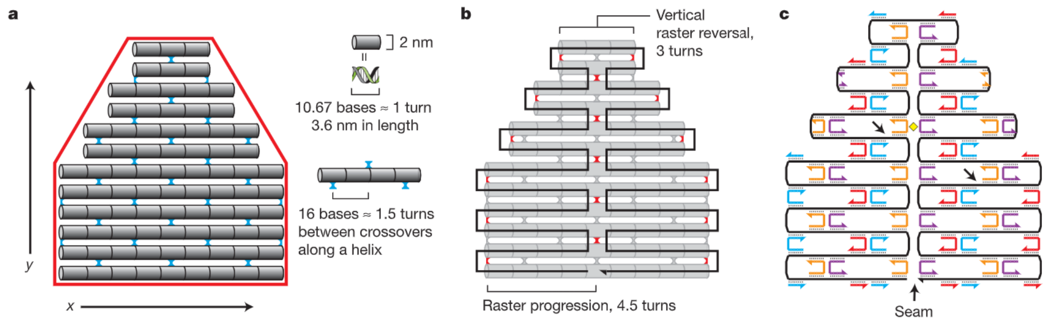


Figure 1.6: Design of DNA origami. (a) A shape (red) approximated by parallel double helices joined by periodic crossovers (blue). (b) A scaffold (black) runs through every helix and forms more crossovers (red). (c) Most staples bind two helices and are 16-mers. This figure is adapted from (Rothemund, 2006).

we will be using two types of square DNA origami – single-layer and double-layer. These structures will be further explained in Chapters 2 and 3.

Atomic force microscopy (AFM)

The atomic force microscope is a scanning probe microscopy instrument that was first introduced in a 1986 paper (Binnig, Quate, and Gerber, 1986). It is useful for imaging DNA origami, as it has nanometer-scale resolution and is gentle enough to keep the origami intact. There is a 2nm tip on the end of a cantilever that hovers above the imaging surface, and a laser that reflects off of the tip. We most frequently use tapping mode, in which the tip oscillates up and down above the imaging surface at its resonant frequency. When the tip encounters origami, it is deflected such that the oscillation is dampened. The tip must move upwards, away from the origami, in order to regain its original oscillation amplitude. The force deflection is then translated into a height measurement, so the image is reconstructed from the force deflection data.

Tapping mode is preferred over contact mode for use with origami, because as a soft polymer it is prone to being destroyed by the tip. Tapping mode minimizes the amount of time that the tip is in contact with the origami. It is still possible to damage the origami, however, if the tapping force is set too high.

Chapter 2

TOWARDS A MAZE-SOLVING DNA ROBOT

2.1 Introduction

In biology, molecular machines are essential in order to support the functioning of a cell. They have evolved to perform specialized mechanical tasks, often in response to environmental stimuli. Two well-known examples are kinesin, which transports molecules around a cell on microtubule tracks, and ribosomes, which translate RNA into proteins. We take inspiration from these biological molecular machines in considering the possibility of building our own analogous machines, called molecular robots. We ask the question - if we were to design molecular robots out of synthetic biological materials, could we design them to perform as complex of tasks, or more complex tasks, than naturally occurring molecular machines?

It is useful to first consider a strategy for developing the platform of DNA robots. If we were to develop a toolbox of simple, modular skills, then we could compose them together in different ways to build robots with particular, and increasingly complex, functionality. Currently, the barrier to building DNA robots with the complexity that we hope to achieve is that the library of algorithms and building blocks available to construct them is not yet large enough.

Some of the earliest DNA walkers walked via the addition of fuel strands that bind the legs to the track and release them (Shin and Pierce, 2004), or via ATP hydrolysis (Yin et al., 2004). Subsequent robots were developed that demonstrated some combination of speed, autonomy, bidirectionality, processivity, and reusability (Omabegho, Sha, and Seeman, 2009; Lund et al., 2010; Wickham et al., 2012). An example of a fast, bidirectional, autonomous walker is the cartwheeling robot, which binds to subsequent tracks by opposite toeholds each time, resulting in a head-over-heels movement (J. Li et al., 2018).

Clamons, Qian, and Winfree, 2020 explored the theoretical capabilities of surface CRN's, which can exploit spatial organization to achieve interesting functionalities. Using simple instructions, they are, for example, able to emulate cellular automata and produce swarm robotic behaviors. These kinds of behaviors would require robots to have additional skills beyond walking around a track.

There have been a few experimental demonstrations of molecular robots that are able to perform tasks, such as a robot that can collect cargo from multiple two-state devices to produce different products, like an assembly line (Gu et al., 2010). Another robot, the cargo-sorting robot, diffuses around a surface, picks up randomly distributed cargo, and sorts them into bins based on their identities (Thubagere et al., 2017). These are both examples of the potential for molecular robots to be functional, and particularly the cargo-sorting robot was designed in a modular way, such that its parts – recognizing the cargo, picking it up, and dropping it in the correct location – could be easily composed together with future robot designs.

We can take inspiration from other phenomena in nature to identify useful skills that a DNA robot could have. Ants are often called a superorganism because of the way that they organize. They are always found in colonies, and they are highly specialized in the types of labor each member of the colony performs. When foraging, a scout searches for food sources on its own. When it finds something of interest, it leaves behind a pheromone trail that other ants can follow to reach the same source. The ants follow simple algorithms to perform sophisticated tasks.

Exploiting this strategy, we aim to add a new algorithm for maze-solving and a new skill of reversible track modification to the toolbox. Here, maze-solving refers to the capability of finding and marking a direct path from the entrance to the exit in a complex maze-like environment. One could imagine that after the path is marked, a subsequent cargo-carrying robot could efficiently transport molecular cargo to or from the end of the path, similar to the way ants behave while foraging.

There have been theoretical explorations of a maze-solving molecular robot. A molecular spider with deoxyribozyme legs traverses a track of DNA strands, which it may cleave as it does so. Once cleaved, the track strands are shorter and thus if the spider were to revisit those tracks, it would spend a shorter amount of time there than a longer, uncleaved track. Therefore, the robot is biased towards visiting tracks that it has previously not visited (Stefanovic, 2012). This work establishes a theoretical basis for one simple algorithm that could underlie such a robot.

On origami, a domino-like system has been demonstrated that can find all possible paths in a maze with equal probability of following any path at each branch point. Thus if there is one branch point with three branches but only one leading to the destination, only one-third of the origami would exhibit the correct path, while the other two-thirds would illustrate the paths following the two erroneous branches (Chao et al., 2019). This is an effective way to enumerate all possible correct and incorrect paths of the maze, but does not allow for identification of just the correct path. A molecular robot, on the other hand, would be able to carry out an algorithm for finding just the correct solution for every maze it encounters.

In this section we explore mechanisms that help us understand the kinds of strand displacement reactions on a surface that would be necessary in order to build such a robot. Though we do not show a successfully functioning maze-solving robot, we hope that this analysis will guide future investigation into and understanding of four-way strand displacement reactions on a surface.

2.2 Prior work

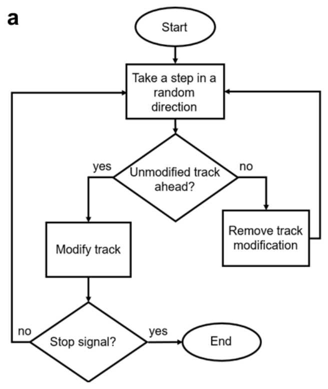
The idea of a maze-solving robot was conceived by my advisor Lulu Qian. She developed a three-way strand displacement design, which was simplified by Philip Petersen, a former graduate student in our lab. Philip also performed simulations showing that DNA origami tile arrays could be leveraged to create stochastic mazes, and estimated how the time that the robot will take to solve a maze scales with the maze size. Together with Grigory Tikhomirov, a former postdoc in the lab, they performed preliminary experiments on the design, testing the walking behavior of the robot on tracks of varying length as well as AFM imaging the track the robot would walk on. Some of these results were presented as a poster at the 20th International Conference on DNA Computing and Molecular Programming (Petersen, Tikhomirov, and Qian, 2014).

A simple algorithm for solving mazes

In order to implement a maze-solving robot, they began with a simple algorithm that would be possible to engineer molecules to execute. The robot performs a random walk and keeps track of the path that does not lead to a dead end. The robot takes a step from the start site; if it hits a dead end, it backtracks and tries another path; and so on, until it reaches the stop site. The robot should not just solve the maze, but also keep track of the solution it has found, so the robot modifies the track on a forward step and removes the track modification on a backward step. This algorithm is illustrated in Figure 2.1.

Desired functions of the robot

The robot was designed with several properties in mind that it should exhibit. First, it should be autonomous, meaning that it does not need active guidance beyond the initial triggering step. This simplifies the experimental process, as well as allows



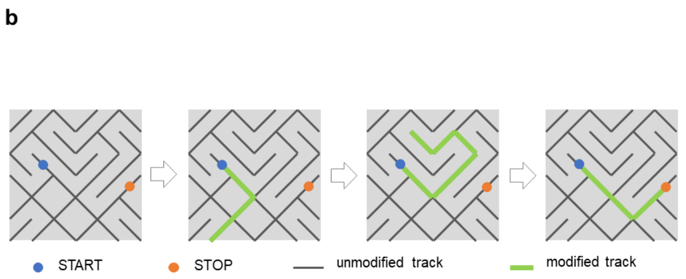


Figure 2.1: A simple maze-solving algorithm. (a) Logical flow chart of the algorithm. The robot performs a random walk; if it reaches a location with unmodified track, it modifies it, and vice versa, until it encounters the stop signal. (b) An illustration of the algorithm. The robot leaves the blue start site and modifies the black track to become green track as it walks. If it reaches a dead end, it backtracks, removing the modifications, and tries another path, until it reaches the orange stop site.

for the robot to be used in environments that could not be easily manipulated. Secondly, it should be bidirectional, meaning that it should be able to take a forward or backward step. An unbiased random walk would allow the robot to solve the maze most quickly, as it would prevent the robot from moving too slowly while also preventing it from getting stuck for too long at a dead end. To achieve an unbiased random walk, the forward and backward reaction rates should be balanced. Thirdly, the robot should be able to mark its path by modifying the track in the forward direction and removing the modification in the backward direction. This way, if it chooses an incorrect path at a junction or hits a dead end, it can retrace its steps and the incorrect path data is erased.

2.3 Initial design

The first design we worked on was a bipedal robot that traverses a track made up of hairpin structures (Petersen, Tikhomirov, and Qian, 2014). It walks via three-way toehold-mediated strand displacement. The mechanism is illustrated in Figure 2.2. As the robot walks, it binds to the blue or red foot toehold and with the leg domain displaces the hairpin open. The open hairpin then initiates a second strand displacement to form a connected bridge-like structure and release the bound foot to take another step.

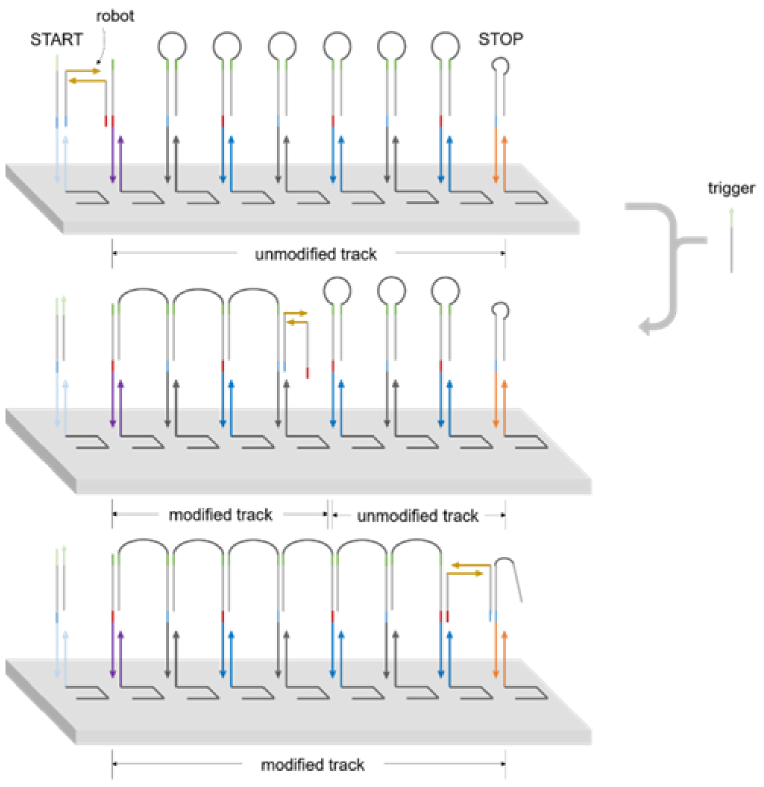


Figure 2.2: The initial robot design. An illustration of the robot design. The grey rectangle represents the origami. The black "c"-shaped lines represent staples, and the vertical lines extending from them are staple extensions to which the hairpins are bound. The robot is comprised of two strands that are bound by a long body domain, shown in gold. It has two unique foot toeholds, blue and red. The top image shows the robot and track before being triggered. The middle image shows the robot in the midst of traversing the track. In the final image, the robot is irreversibly bound to the stop location.

For an unbiased random walk, we would expect

$$n = d^2$$

where n is the number of steps it will take to reach the end of a track of length d. Experimentally, on odd-numbered length tracks, the completion levels differ for 3-, 5-, and 7-staple tracks, but are the same for 7- and 9-staple tracks. The kinetics fluorescence data for odd-numbered tracks is shown in Figure 2.3b. Additionally, the 4-, 6-, and 8-staple tracks all have very low and comparable completion levels, suggesting that the robot is not actively walking on those tracks. If the system were working as designed, we would expect to see a longer half-completion time as the tracks grow in length, but the same final completion level, as indicated in Figure 2.3a.

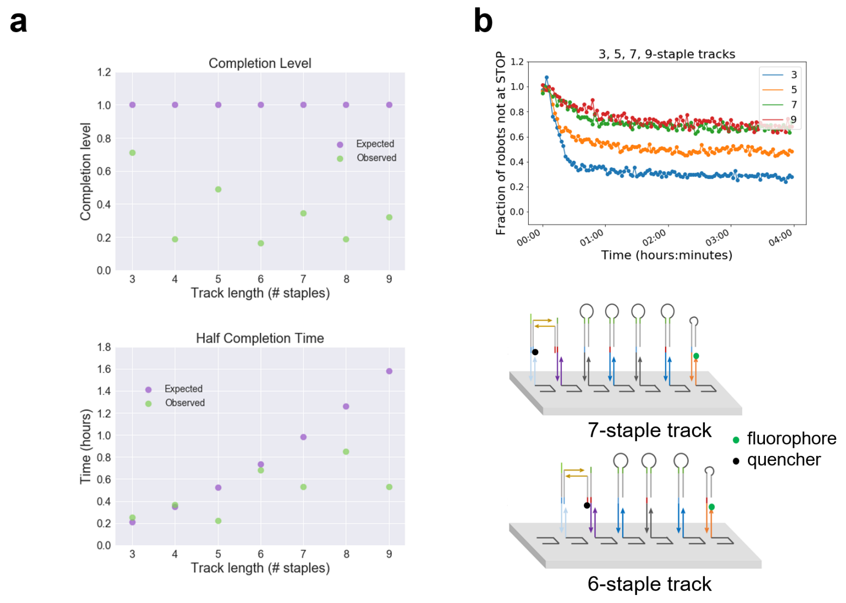


Figure 2.3: Simulations and data of walking behavior. (a) Top, the expected completion level vs. the observed for tracks of various lengths. Bottom, the expected half completion time vs. the observed for tracks of various lengths. (b) Top, representative fluorescence kinetics data below showing the fraction of robots to reach the stop location on odd-numbered length tracks. Middle, an illustration of a 7-staple track representative of odd-numbered length tracks. Bottom, an illustration of a 6-staple track representative of even-numbered length tracks with corresponding even-numbered track length data.

To debug the robot behavior, we simplified the system such that a single backbone strand of DNA, rather than a DNA origami, was holding together the two start locations, the track and the stop location. This resulted in an 11-stranded complex. However, this complex proved difficult to purify by PAGE gel purification, as it was too large and had too much secondary structure to successfully travel through the gel matrix. A larger gel matrix – e.g., an agarose gel – would not be fine-grained enough to separate the 11-stranded desired complex from, for example, a 10-stranded complex that was missing a strand.

Given the difficulty of designing and purifying a simpler system by which to debug this robot, we decided to redesign the robot in such a way that it would be easier to investigate its behavior when faced with inconclusive data on origami.

2.4 Redesigning the robot

Considerations in redesigning the robot

While debugging the initial robot design, we made several observations that helped inform a new design. Firstly, the robot is quite slow compared to other robots, such as the cartwheeling robot that moves at a speed of $7.2*10^{-1}s^{-1}$ (J. Li et al., 2018). The cargo-sorting robot is also much slower than the cartwheeling robot, moving at a speed of $3.5*10^{-3}s^{-1}$ (Thubagere et al., 2017). The initial maze-solving robot design and the cargo-sorting robot share the property that during a single step, the loop size formed by the robot and the track gets smaller as a single-stranded region becomes double-stranded, incurring an entropic cost. In the case of the cartwheeling robot, the loop size does not change during the course of a walking step, and thus there is no extra entropic cost incurred. This extra cost could be contributing to the slow speed of the maze-solving robot, and thus should be avoided in the next design.

Secondly, as the robot walks and modifies the track by opening up the hairpins, it forms a connected bridge-like structure. There is an entropic cost of forming these loops, and in this case, because the structure is fully connected, the entropic cost of each step is dependent on the number of steps previously taken, rather than being constant with each new step. This extra cost per step, compounded by all previous steps taken, could also have contributed to the robot's slow speed, and could explain why as the track gets longer, as in the 7- and 9-staple cases, the speed and completion level plateau.

Finally, the track hairpins are attached to the track by staple extensions because they contain several long domains. If the hairpins were extended directly from the staples, the number of nucleotides in each strand would exceed the synthesis limit of the typical products we order from IDT. This raised track design is prone to spurious interaction because the reachability of each hairpin is greater than if it were attached directly to the origami. This could result in the robot being able to skip steps to reach the stop location, as seems possible in the even-numbered track cases. There are two more related issues as well. First, adding single-stranded regions in between the double-stranded domains to allow more flexibility and a less constrained geometry could increase the speed of the robot, as strand displacement requires some reconfiguration of the complexes involved. However, there is little room to add this flexibility because the hairpins are already so long. The second issue is that as strands approach the synthesis limit IDT has set, they are much more prone to synthesis errors.

The redesigned robot

The redesigned robot is a single strand of DNA with one long leg domain and two short foot domains, foot toehold T_f. It walks along a track that contains another toehold, the bridge toehold T_b. The track is made up of double-stranded complexes that are extended from staples whose 3' and 5' ends meet at the same nick location on the origami scaffold. An example is circled in red in Figure 2.7c and the track configuration is discussed further in Section 2.5. The robot walks along the track via a four-way branch migration mechanism which converts the unmodified track into modified track. The walking behavior of the robot is illustrated in Figure 2.4, while the mechanism of the four-way strand displacement in solution is illustrated in 2.5a.

In this design, the size of the loop that is formed does not change during a reaction step, as shown in Figure 2.4b. The loops are not connected to each other, as shown in Figure 2.4c. Therefore, the entropic cost of each step is constant regardless of how long the track is. Finally, as shown in Figure 2.4d, the tracks are shorter and therefore can be directly extended from staples. They can no longer reach beyond their neighbors to spuriously interact with other track locations, and there is room to modify the design by adding nucleotides. All design considerations that were made following the initial design are met in the redesigned robot.

2.5 Experimental implementation

The robot

Given the geometry of the track locations with the track, particularly in the modified form as shown in the bottom panel of Figure 2.4, we chose to make the length of the double helix span an integer and a half number of turns. Thus in the modified form the linkers would be pointing in the same direction on oppositely oriented strands once the loop formed. We also considered the constraint that the robot would always be untethered from the track, and therefore the leg domain must be long enough to be stably bound as it would be in a free-floating complex. Thus we chose the length of the leg domain to be 20 nucleotides while we made T_b 3 nucleotides and T_f 3 nucleotides. Together, the 26-base pair double-stranded domain would be about 2.5 turns of the helix. We also considered how long the linker lengths should be given the spacing of the tracks on the origami. The distance between track locations, L_t, is 10.5nm, as shown in Figure 2.6a. The length of the 26-base pair helix, L_h, is 8.84nm. To estimate the necessary linker length, we imagined a triangle on either side of the double-stranded domain, as shown in Figure 2.6a, and solved for x and

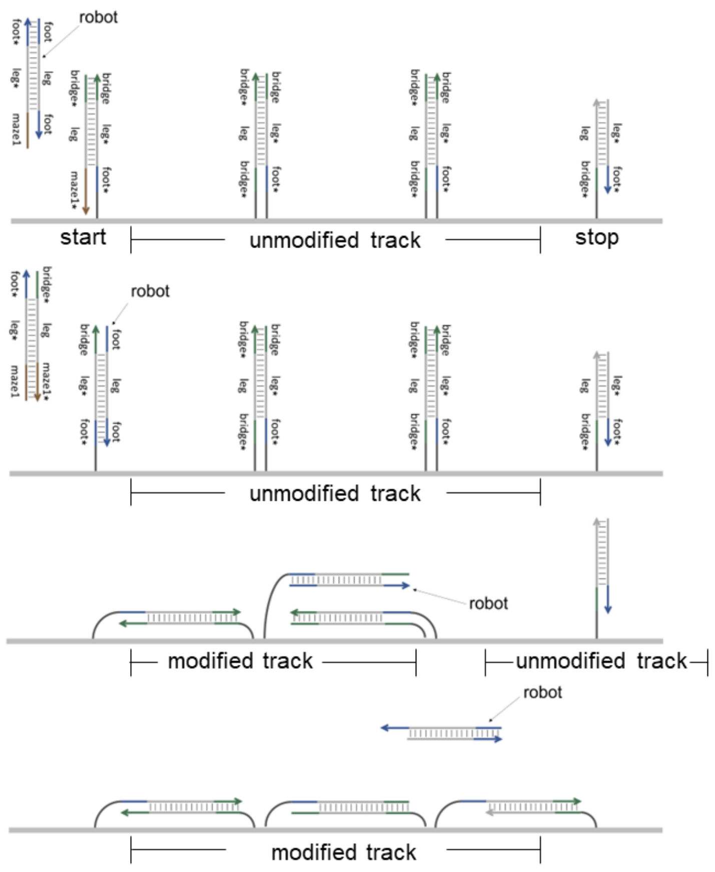


Figure 2.4: An illustration of the redesigned robot. The grey rectangle represents the origami. The vertical lines attached to the origami are staple extensions. The robot is comprised of one long leg domain and two short foot domains. The top image shows the robot contained in a free-floating complex. The next panel shows the robot localized to the origami via four-way strand displacement. In the third panel, the robot has taken one reversible step using the blue foot and green bridge toeholds and could take a step forward or backward. In the bottom panel, the robot has reached the stop complex and been released from the maze.

y to get z, the linker length. $x = \frac{1}{2} * (L_t - L_h) = .83nm$, while $y = 2nm$, the width of a double helix. Solving for z, the linker length, we get $z = 5.03$. Thus we made each linker 5 nucleotides. We could have chosen to make the linkers slightly longer, for example 6 nucleotides, but we considered that .43nm is the average length of a nucleotide in a single strand, and can stretch to be longer than that depending on the

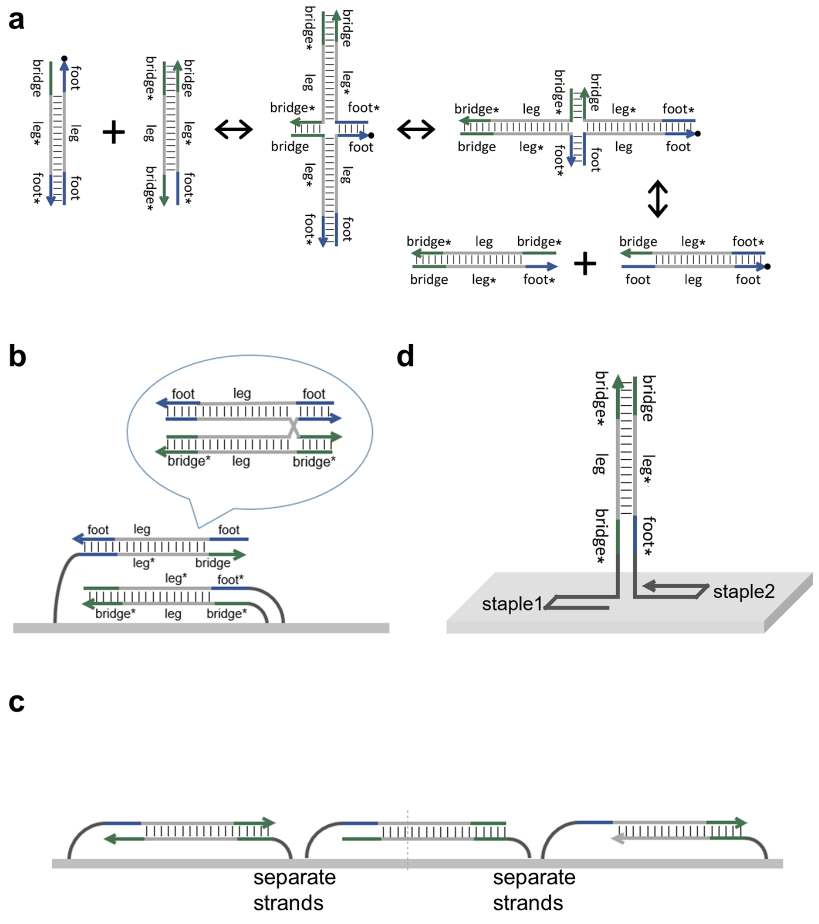


Figure 2.5: Details of the robot design. (a) An illustration of reversible four-way strand displacement in solution. (b) A schematic showing that the loop size stays constant during the entire four-way strand displacement step. The parallel stacked-X form of the Holliday junction is also shown. (c) A schematic showing that the loops that are formed are not connected to each other. (d) A schematic showing that the track strands extend from staples on origami.

system configuration. We will discuss experiments in which we change the length of T_b - in those cases, as we increase T_b, we decrease the linker length by the same number of nucleotides such that the leg domains of two strands making up a track complex are evenly spaced from the origami surface.

The track design

The track upon which the maze is built is a double layer DNA origami tile, as shown in Figure 2.7. Double layer origami provides structural rigidity over single layer origami. Because in single layer origami all the helices point in the same direction,

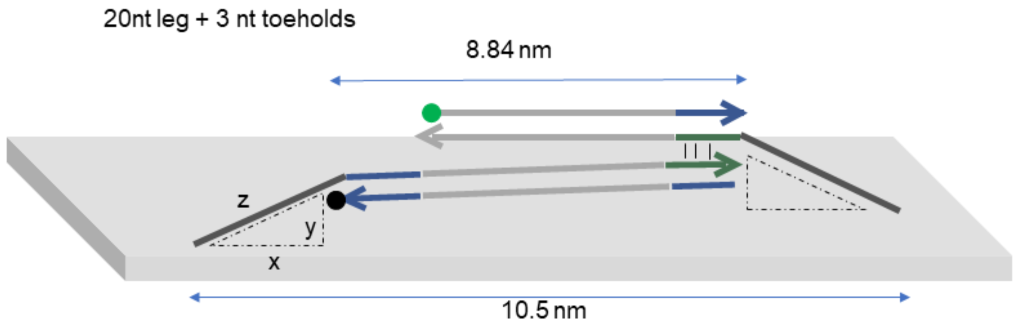


Figure 2.6: The geometry of the robot. (a) A diagram showing the geometry of the loop formed between two track locations, as well as the approximation used to calculate the lengths of the single-stranded linkers that tether the complex to the origami surface.

the tile can fluctuate in solution in ways that allow, for example, opposite corners of the tile to come into close proximity. In that event, the robot could spuriously interact with a track on the other side of the origami that it would not have been able to reach if the origami had been flat, effectively "jumping." In contrast, in double layer origami, the perpendicular helices, shown in Figure 2.7b, prevent the origami from fluctuating as easily along the z-axis. Therefore, it should not be possible for the same kinds of spurious interaction to occur as on single-layer origami.

There are staple junctions on the top layer of the origami, shown in Figure 2.7c. The track is created by extending certain staples at their 3’ or 5’ staple junction.

There are 212 staples that hold the M13mp18 scaffold in the square double-layer shape. The staples vary in length from 30 to 48 nucleotides. The scaffold is mixed with a 5x excess concentration of the staple strands and annealed from 90°C to 20°C at 6 sec/.1°C. The annealing protocol takes about 90 minutes. At the peak temperature of 90°C, all bonds are broken between strands, so all strands are single-stranded and lose their secondary structure. As the temperature is slowly lowered, the strands find their most energetically favorable state, which is as fully bound as possible. Therefore, this state should correspond to the desired structure. To remove excess staples after annealing, the origami is filtered in 100K filters.

Annealing the track

While all track locations have both strands bound to the origami, the start and stop locations both contain strands that are not bound to the origami. This is necessary in order to make the first step, in which the robot localizes to the maze, and the last step,

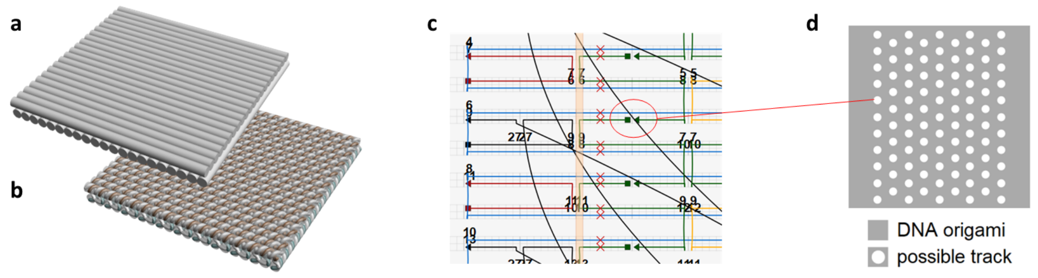


Figure 2.7: Double-layer DNA origami. (a) A representation of double layer origami, with DNA helices shown as rods. The helices in the lower layer are perpendicular to those in the upper layer, providing structural stability. (b) A representation of double layer origami, with DNA helices shown in each rod. (c) A portion of a cadnano schematic of the origami, with scaffold shown in blue, edge staples in red and black, and internal staples in green. A scaffold junction is circled in red. At these junctions, the 3' or 5' end of a staple, shown as an arrow or a square respectively, can be extended to become a track location. (d) A bird's-eye view of the surface of the origami, with each white circle indicating a staple junction that is a possible track location.

in which the robot leaves the maze, irreversible. When the strands comprising the start and stop complexes are present during annealing, there is a possibility that the track could anneal in its modified form, releasing the strands that are not bound to the origami. This is because there is an entropic gain of +2 free-floating molecules. However, there is also the entropic loss of the track molecules forming loops. It is difficult to quantitatively compare this gain and loss and determine which would prevail.

As the track gets longer, the entropic loss of forming loops will increase and thus it becomes more likely that the unmodified track is favored. As our initial experiments were with short tracks, to avoid this potential issue we developed a protocol to insert the start and stop complexes into the origami after annealing the rest of the track.

Here we refer to either the start or stop complex as the insertion complex to indicate the generality of these results. To perform this experiment, we annealed the origami with a staple missing. We separately annealed the complex containing the staple extension and the strand without a staple. As shown in Figure 2.8a, the complex was then added to the origami and allowed to hybridize for 16 hours. The positive control, shown in shown in Figure 2.8b, was a staple extension annealed with the origami, to which a complementary quencher strand was then added in excess and allowed to hybridize.

At an insertion complex concentration of 1x compared to 1x origami, where 1x=3nM, we saw 20% insertion after 16 hours, but with a 10x insertion complex concentration, we were able to achieve 80% insertion into the target origami after 5 hours. These results are shown in Figure 2.8d. Increasing the concentration to 20x did not make a big difference to the kinetics, and achieved the same 80% completion level as 10x.

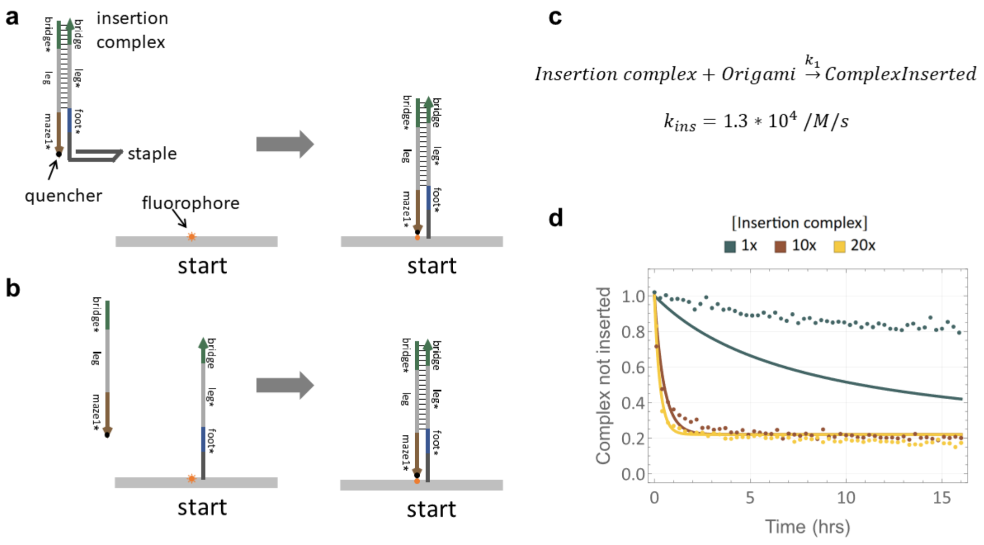


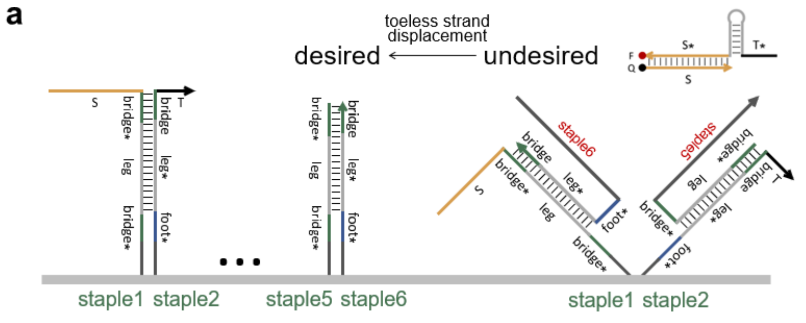
Figure 2.8: Inserting complexes into the origami post-annealing. (a) A diagram of the experimental design, in which a complex containing a staple is hybridized to the origami. (b) The positive control of the experiment, in which the staple with its extension is annealed with the origami, and a complementary quencher strand is added afterwards. (c) The irreversible reaction and reaction rate used in the model. (d) Fluorescence kinetics data showing the insertion efficiency when 1x, 10x, and 20x complex are added relative to the origami concentration, which is at 1x=3nM.

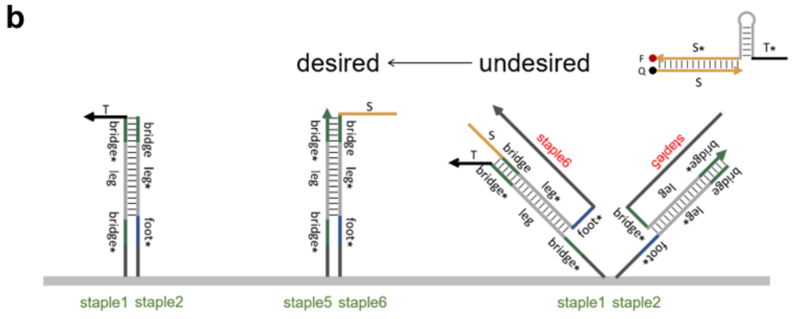
We model the reaction using mass action kinetics (Soloveichik, 2009). Here, there is a disagreement between the model and the simulation in the 1x insertion complex case. This could be the result of synthesis errors. In future experiments, it would make sense to test other concentrations in between 1x and 10x to determine at what point the thresholding effect is overcome, and thereby form a better hypothesis for what could be causing it. There is also generally a low reaction completion level for the 1x case. We hypothesize that this may be due to repulsion between the complex and the large, negatively charged DNA origami. This effect would be especially pronounced with double-layer origami, and may be overcome with large enough complex concentrations. We also tried to insert the 1x concentration at 25°C, 30°C, and 35°C, and did not see a significant difference in the result, further supporting this hypothesis as the temperature should not affect the repulsion unless the helices of the origami were to bow significantly, or the origami were to fall apart altogether.

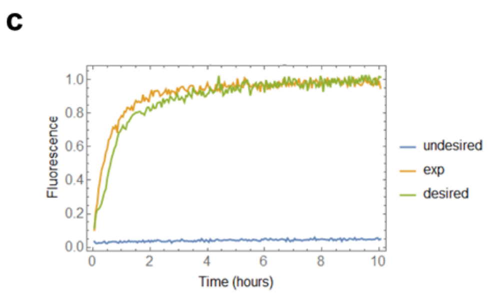
There is another type of incorrect annealing that could occur: because there are two types of track locations that repeat with identical domains, there is a possibility that a strand from a different track location could bind to the wrong partner during annealing. However, this would mean that one of the strands at that track location would not be bound to the origami. Due to the entropic gain of an additional free-floating strand, the correct staple should bind at the correct staple location and strand displace away the incorrect track strand. In other words, the entropic gain of freeing the wrong partners would cause it to undergo toeless strand displacement to become the desired case. Using a fluorescent reporter method, we performed an experiment to determine whether the track anneals as desired, as shown in Figure 2.9a. The data in c shows that the track is annealing as desired, although there are two track locations per origami, so it indicates that 50% of the track is annealing as desired or correcting itself before we can detect otherwise. We then performed the opposite experiment, in which we measured the presence of incorrect track structures, as shown in Figure 2.9b.The data in d shows three different experimental conditions: top left, our normal annealing conditions at 6 sec/.1°C; top right, a slower anneal, at 12 sec/.1°C; and bottom, the standard anneal of 6 sec/.1°C plus a 24-hour hybridization period at room temperature that should allow the track to correct itself if it annealed incorrectly. The mechanism by which it should correct itself is that because of the unimolecular nature of staple1 and staple2 in Figure 2.9a, if staple5 and staple6 were to bind as shown on the right, staple1 and staple2 should perform a zero-toehold strand displacement to form the desired structure. The standard annealing condition agrees with the previous experiment, in that about 50% of the track is annealing correctly. While the slower 12 sec anneal helps, the most effective method is allowing for the correction period at room temperature after the initial standard anneal.

Direct vs. indirect reporters

There are two kinds of reporting methods used in the experiments discussed in this chapter. The first, a direct reporter, entails a quencher molecule on one end of the robot and a fluorophore either on the track or on the stop complex, as shown in Figure 2.10a. An indirect reporter is triggered downstream of the reaction of







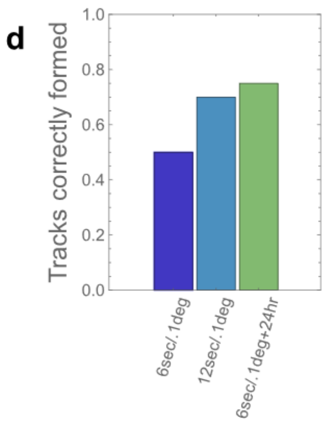


Figure 2.9: Annealing the track in one pot. (a) The track could take on two possible configurations when more than one of the same track type is present: the undesired case at right, which shows the track strands bound to the wrong partners, or the desired case at left. (b) The experimental design to measure the incorrectly formed track structures, rather than the correctly formed ones. (c) Fluorescence kinetics data from the experiment in (a). Because there are two reporter locations per track, the experimental sample labeled "exp" indicates that about 50% of the track is annealing correctly. (d) We tried three different annealing protocols: first, the standard anneal of 6sec/.1°C, which gave us about 50% correct track formation, agreeing with the results in (b). We tried a longer anneal of 12sec/.1°C, 70% correct formation, and 6sec/.1°C plus a 24-hour hybridization period at room temperature, 75% correct formation.

interest, as shown in Figure 2.10b.

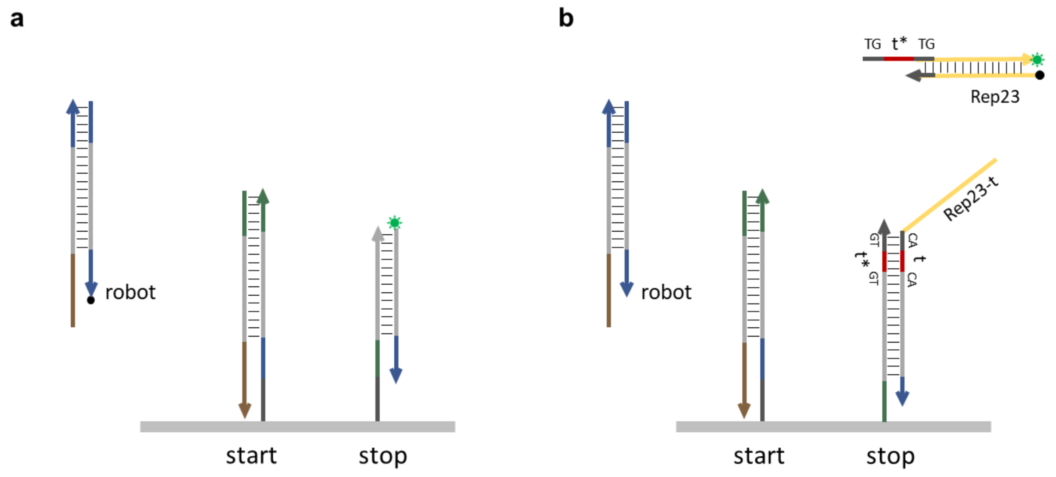


Figure 2.10: Direct vs. indirect reporters. (a) A direct reporter, in which the quencher (black circle) is on the robot strand and the fluorophore is on the stop (green circle). (b) An indirect reporter. In this case, the robot reacts with the stop complex to reveal a toehold that allows the resulting complex to trigger the reporter, Rep23. The grey leg domain here is modified to match the clamp domains that surrounded t, the universal toehold that triggers Rep23. Therefore, leg and leg* no longer follow the 3-letter code.

The advantage of a direct reporter is that the system is as simple as possible, as there are no extra reactions, such as the triggering of the reporter. However, each time we make a design change in the robot or the stop complex, we must also order new fluorophores and quenchers. Not only does this become costly, but there is the practical limitation that it takes IDT 2 weeks to produce and deliver fluorophore- or quencher-modified strand. Therefore, iterating through new designs is expensive and slow.

On the other hand, an indirect reporter is universal, in that any system can be designed to trigger it. Therefore, as the design changes, the reporter can stay the same. However, based on which sequences are needed to trigger the reporter, this can sometimes complicate the design of the system or introduce single-stranded domains that could spuriously interact with the other complexes. In the example shown in Figure 2.10b, the end of the grey leg domain is modified to match the clamp sequences needed around the universal toehold that triggers Rep23, which causes them to no longer follow the 3-letter code. Another effect of the indirect reporter is that the overall kinetics that we see will also include this extra reaction, though it is often not the rate-limiting step.

This project began with direct reporters, but as we made design changes, we transitioned to an indirect reporter. The question of which type of reporter to use will depend largely on the type of research project, the likelihood of design changes that will affect the modified strands, and the timeline that the researcher expects to be on.

Visualizing the track on AFM

Besides fluorescence kinetics, our other readout method is via the atomic force microscope (AFM). We can use the AFM to visualize the origami. The main motivation with this project is to be able to distinguish the unmodified from the modified track such that we could use a visual readout to show us the solution to the maze that the robot has found. We needed to verify that it would be possible to distinguish between the two types of track. Imaging on double-layer origami is more challenging than on single-layer, as it is softer and thicker, and therefore surface modifications are not as clear.

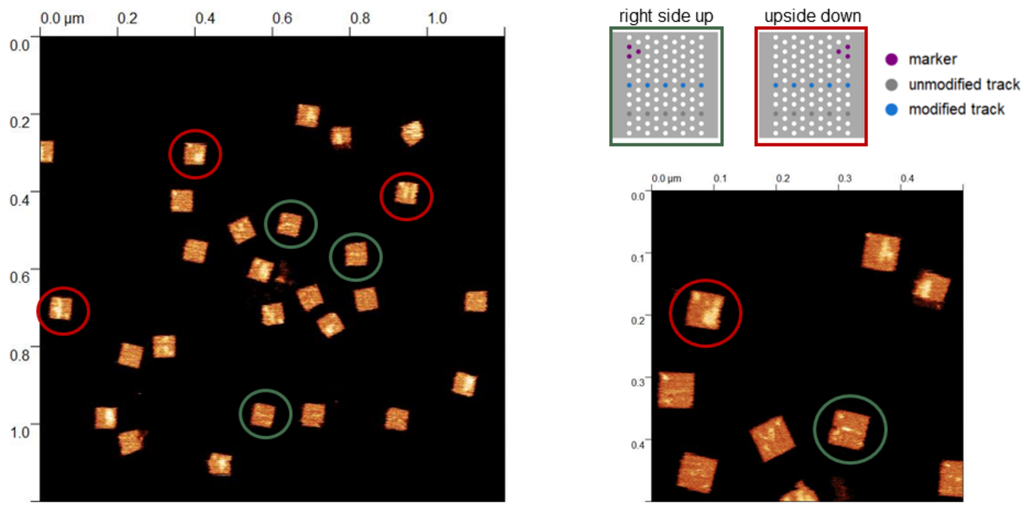


Figure 2.11: Visualizing the track on AFM. A schematic showing the origami design is shown in the top right. The marker in the top corner of the origami helps us differentiate origami that has landed right side up (outlined in green) from origami that has landed upside down (outlined in red). We are able to distinguish the modified from unmodified track on origami that have landed right side up, but not on origami that has landed upside down.

We answered this question by putting a line of track that has unique leg domains forcing it to be unmodified and a line of track that has unique leg domains forcing it to be modified on the same origami. Bird's-eye view schematics of the design are shown in the top right of Figure 2.11.

The AFM is a physical imaging instrument, so it is possible for the cantilever tip to push surface modifications on origami out of the way as it rasters. Because of this, we hypothesized that unmodified track, fixed at one point, would be pushed out of the way more easily and the modified track, fixed at two points, would not. This could result in the two types of track being distinguishable in an AFM image. The one caveat, however, is that when origami land upside down, the surface modifications are less likely to be pushed out of the way, possibly because the tile itself holds them in place.

We image DNA origami on a mica surface, and it is possible for it to land right side up or upside down. Biasing which way it falls would require modifying the surface in such a way that might interfere with the robot behavior. Instead, we added a dot to one corner of the origami so that we could tell whether it had landed right side up or upside down on the mica.

When the origami lands upside down, we are unable to distinguish between the unmodified and modified tracks. However, when the origami lands right side up, we are able to see a sharp line of the modified track and a blurry or nonexistent line of the unmodified track, as shown in Figure 2.11.

2.6 Irreversible four-way strand displacement on a surface

Investigating reaction rates

Most of the walking behavior is reversible four-way strand displacement, with the exception of the first and final steps, which are both irreversible four-way strand displacement. We measured both the reversible and irreversible four-way strand displacement rates in solution and on origami for a robot with a 3-nucleotide foot toehold, T_f, 3-nucleotide bridge toehold, T_b, and 20-nucleotide leg domain. We used reaction rates of four-way strand displacement in solution published as part of a doctoral dissertation as benchmarks for our measured rates in solution (Dabby, 2013). We also estimated some unimolecular reaction rates based on those bimolecular rates. We will discuss these estimates further in the next section.

In Dabby, 2013, the four-way strand displacement reaction is modeled by two reaction rates, k_1 and k_2, shown in Figure 2.12a. k_1 refers to the rate of successful initiation of four-way strand displacement, which is dependent on the two relevant toeholds, whose lengths are referred to as m and n. k_2 refers to the rate of four-way branch migration, which should not change for any m and $n > 2$. In Dabby, 2013,

k_2 for m and $n > 2$ was found to be $1.5 * 10^{-3} s^{-1}$. They found the bimolecular rate constant k_1 for $m = 4$ and $n = 2$ to be $56 M^{-1} s^{-1}$. Rather than k_1, we will call this parameter k_i, for irreversible. We measured the bimolecular rate constant k_i for our system in solution, in which $m = 3$ and $n = 3$, to be $70 M^{-1} s^{-1}$, which is comparable. We increased $T_b = 4$ and used the rate constant k_1 for $m = 4$ and $n = 4$, which in Dabby, 2013 is $770 M^{-1} s^{-1}$. We found our rate constant to be $800 M^{-1} s^{-1}$, as shown in Figure 2.12a. Finally, we further increased $T_b = 5$ as shown in Figure 2.12b and used the rate constant k_1 for $m = 6$ and $n = 2$ for comparison. The rate from Dabby, 2013 is $9.4 * 10^3 M^{-1} s^{-1}$, while ours is $2.5 * 10^3 M^{-1} s^{-1}$. While the rates agree in order of magnitude, we attribute the discrepancy to the slightly different toehold lengths and differences in sequence. We use mass-action kinetics to simulate these reactions (Soloveichik, 2009).

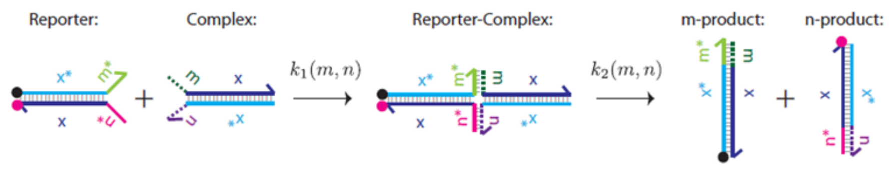
Toeholds	Rate $M^{-1} s^{-1}$	Ref. toeholds	Rate $M^{-1} s^{-1}$	Ref. toeholds	Rate $M^{-1} s^{-1}$
$T_f = 3, T_b = 3$	70	$m = 4, n = 2$	56	$m = 2, n = 4$	.93
$T_f = 3, T_b = 4$	800	$m = 4, n = 4$	770		
$T_f = 3, T_b = 5$	$2.5 * 10^3$	$m = 2, n = 6$	490	$m = 6, n = 2$	$9.4 * 10^3$

Table 2.1: Measured reaction rates compared to reference rates from Dabby, 2013. The rates were different when the two toeholds were swapped, either of which could arbitrarily correspond to either of ours. Thus both are shown for reference.

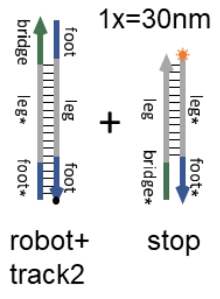
On origami, we used the following reactions to estimate the reaction rate. {1} The $robot : track_{free}$ complex is first inserted into the origami via a hybridization reaction, shown in Figure 2.13a(1)-(2), at the rate k_ins, which was measured to be $1.3 * 10^4 / M/s$ in the insertion experiment. {2} The $robot : track$ complex, now inserted into the origami, hybridizes to the stop complex and initiates branch migration; we call this combined rate k_i, and this state, which we call $robot : track : stop_{ori}$, is shown in Figure 2.13a(3). {3} The $robot : track : stop_{ori}$ complex performs branch migration and dissociates by a single 3-nucleotide toehold (the foot toehold). Though we know this rate is $10^3 s^{-1}$[1], we use the rate of branch migration as the forward rate for this reaction, because it is slower than dissociation and thus rate-limiting. We call the branch migration k_{BM}, as this will be the parameter we tune. We expect that it would change from solution to origami due to changes in geometric constraints on the reaction. There are now two complexes: the origami with a loop formed, called $track : stop_{ori}$, and the free-floating $robot : stop_{free}$ complex, as shown in Figure 2.13a(4).

[1]The formula for calculating the dissociation rate is $10^{6-L}/s$, where L is the toehold length.

a



b



c

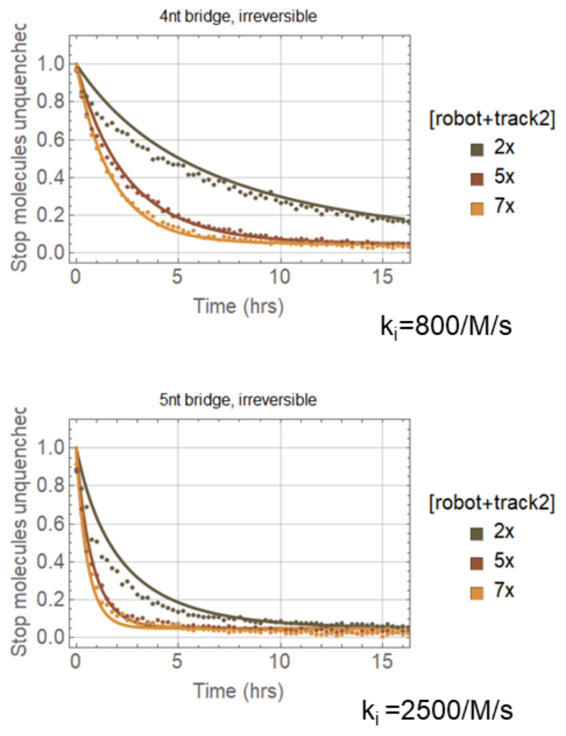


Figure 2.12: Irreversible four-way strand displacement in solution. (a) The model for four-way strand displacement, from Dabby, 2013. Here k_1 is equivalent to our k_i, which we have renamed to indicate the irreversible reaction. (b) A diagram of the irreversible reaction in solution. (c) Fluorescence kinetics data of the irreversible reaction when $T_b = 4$ or $T_b = 5$ nucleotides, respectively. Dotted lines show experimental data and solid lines show simulations.

Chemical reaction network for an irreversible step on origami:

$$\text{robot:track}_{\text{free}} + \text{stop}_{\text{ori}} \xrightarrow{k_{ins}} \text{robot:track\&stop}_{\text{ori}} \qquad \{1\}$$

$$\text{robot:track\&stop}_{\text{ori}} \xrightarrow{k_{\text{i}}} \text{robot:track:stop}_{\text{ori}} \qquad \{2\}$$

$$\text{robot:track:stop}_{\text{ori}} \xrightarrow{k_{BM}} \text{track:stop}_{\text{ori}} + \text{robot:stop}_{\text{free}} \qquad \{3\}$$

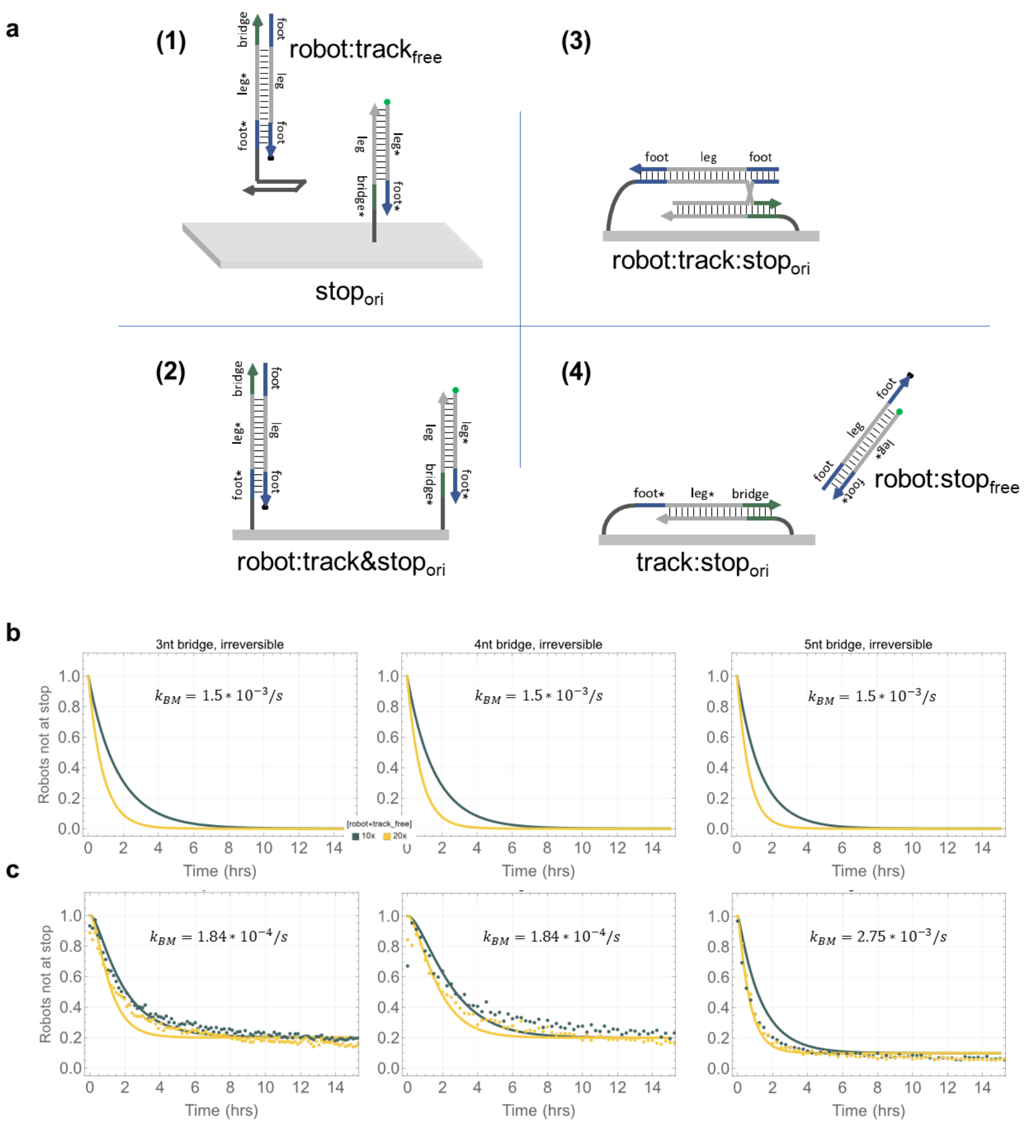


Figure 2.13: Irreversible four-way strand displacement on origami. (a) The states of the reaction. (b) The idealized simulation for $T_f = 3$ and $T_b = 3$, $T_b = 4$ or $T_b = 5$, respectively. (c) Fluorescence kinetics data of the irreversible reaction when $T_b = 3$, $T_b = 4$ or $T_b = 5$, respectively. In these simulations, k_{BM} has been tuned to better fit the data. The completion level has also been tuned to account for the 80% incorporation rate of the stop complex, as noted in the previous section. The completion level for the T_b=5 case had a slightly higher completion level than the expected 80%, and thus was tuned to 90%.

To find the idealized k_i, we started with the bimolecular rates we found in Table 2.1, which matched up well with the rates from Dabby, 2013. Given that we used slightly different toehold lengths than the reference rates, and given that the rates matched well, we used our measured rates to estimate unimolecular reaction rates.

We began with estimating the local concentration of the $robot : track$ complex on origami. We assumed that the length of a base pair is $.34nm$, while the length of a nucleotide in a single-stranded domain is $.43nm$. The leg domain is 20 nucleotides long and the foot is 3 nucleotides long, plus there is a 5 nucleotide single-stranded spacer localizing the helix to the origami. Thus, we calculated the length of the $robot : track$ complex, L_c, to be $23bp * .34nm + 5nt * .43nm = 9.97nm$. The tethered complex can move in any direction except through the origami, so we considered its volume to be the dome that its length is the radius of. We calculated the volume of that dome, v_d, to be $v_d = \frac{1}{2} * \frac{4}{3} * \pi * L_c^3 = 2075.6nm^3$. To find the local concentration, we divided 1 molecule by v_d to get a local concentration, C_L, of $8 * 10^{-4}M$. The unimolecular k_i is given by:

$$k_i^{uni} = k_i^{bi} * C_L$$

Plugging in the rate from Table 2.1, we get a unimolecular rate $k_i = 5.6 * 10^{-2}s^{-1}$. We can similarly calculate these rates for $T_b = 4$ and $T_b = 5$. The calculated rates are shown in Table 2.2.

Toeholds	Calculated Rate s^{-1}
$T_f = 3, T_b = 3$	$5.6 * 10^{-2}$
$T_f = 3, T_b = 4$	$5.79 * 10^{-1}$
$T_f = 3, T_b = 5$	1.64

Table 2.2: Calculated and fitted unimolecular reaction rates k_i for the irreversible reaction on origami.

We can now compare a simulation of the expected system behavior with our experimental results. Because the branch migration rate is the limiting step, we chose to use the idealized k_i and tune the branch migration rate, k_{BM}. The idealized behavior is shown in Figure 2.13b, while in c the k_{BM} and completion level have been tuned to better fit the data. Though the rate $1.8 * 10^-4/s$ fits well for the T_b=3 and T_b=4 cases, the fitted rate for the T_b case is $2.75 * 10^-3/s$, about an order of magnitude faster, and similar to the reference branch migration rate of $1.5 * 10^-3/s$ from Dabby, 2013. We considered several hypotheses for what may be slowing the reaction down in the shorter toehold cases, which we will now explore.

Effect of purified strands

One practical limitation that we face is the quality of the strands purchased from IDT. We generally use unpurified strands, which means that some percentage of the strands in the batch will be truncated to a lesser or greater degree. Our goal is to design systems that are robust to this level of impurity, but it is still informative to see to what extent the sub-optimal behavior that we observe is due to the quality of the strands. Given the method of synthesis that IDT uses, synthesis errors are most likely to be located at the 5’ end of the strand. That would mean that one of the invading toeholds may be affected, which could result in a slowdown of the reaction.

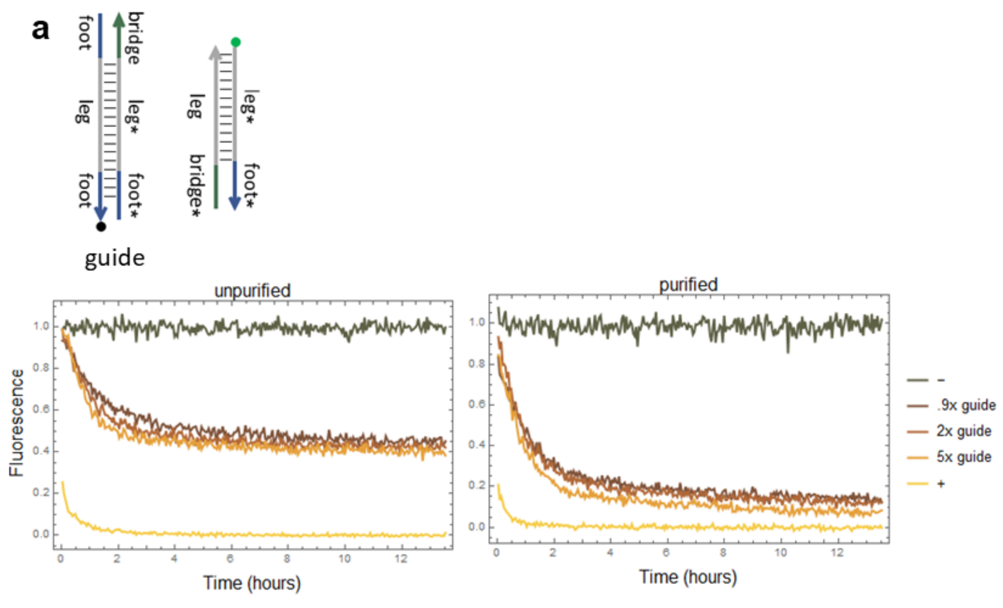
We investigated using PAGE-purified strands, ordered from IDT, on the irreversible reaction in solution. We tried two versions of the reaction: in Figure 2.14a, the guide complex reaction with the start complex, and in Figure 2.14b, the guide complex reacting with the start complex, which then reacts with the stop complex. All four-way strand displacement reactions in these two scenarios are irreversible. In the first case, the speed and completion level of the reaction both improved. In the second case, the only marked improvement was in the .9x guide case, which caught up to the higher concentration cases. If some percentage of the molecules are truncated, then using a higher concentration of the strand would result in more fully formed molecules participating in the reaction. Based on these results, it did not seem likely that we would see an order of magnitude difference in the reaction rate on origami. Therefore, we chose to not pursue experiments with purified strands on origami.

Testing different start locations

Another factor that we considered is whether the geometry of the origami could result in some track locations being more favorable to the reaction than others, and thus resulting in a slower or faster reaction rate. To investigate this possibility, we placed the robot in three different start locations to see if the reaction rates were comparable. The diagram of the reaction set up is shown in Figure 2.15a, while the fluorescent kinetics data is shown in Figure 2.15b. The kinetics and completion levels of the reaction starting in each of the three locations are almost identical, suggesting that the start location does not affect the reaction.

Increasing the local concentration of the start and stop complexes

Another possible roadblock to achieving maximal completion level of the robot is that some tracks are missing a robot, track, or stop complex altogether. Ideally, we would add redundancy into all parts of the track to combat this issue, as in the cargo-sorting robot (Thubagere et al., 2017). However, the strategy of having multiple paths that the robot can take after each step would not work for this design. The robot would be able to move in a circle, i.e. cross back over a previous path,



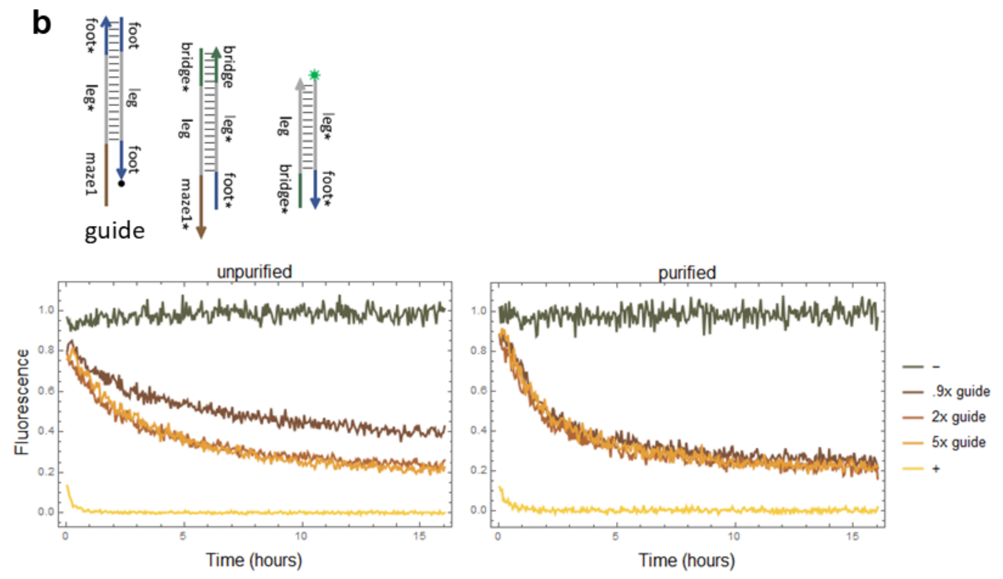


Figure 2.14: Reactions using PAGE-purified strands. (a) A diagram of the irreversible reaction in solution, with data comparing the experiment using unpurified and purified strands. (b) A diagram of the localization reaction plus irreversible reaction in solution, with data comparing the experiment using unpurified and purified strands.

which because of the loop-formation could result in it getting tangled up. Instead, we chose to test this by adding redundancy into just the dock and stop locations, as shown in the schematics in Figure 2.15c.

With an increasing number of robots, the speed and completion level of the reaction

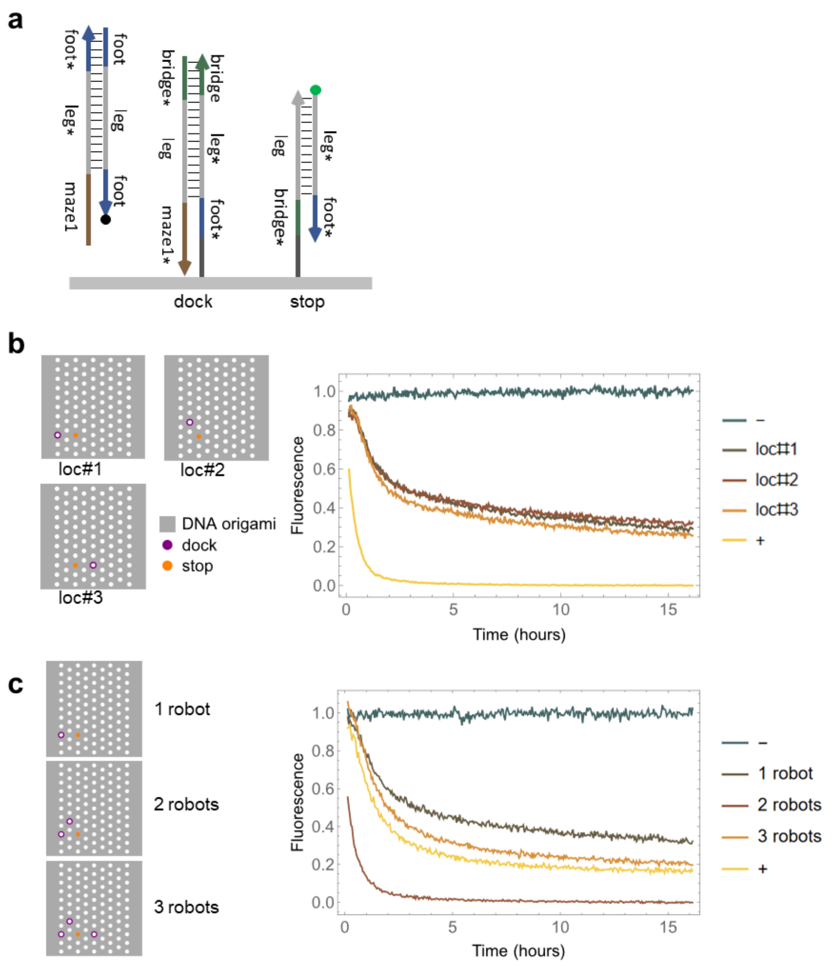


Figure 2.15: Probing the completion level of the irreversible reaction. In all fluorescence kinetics data, the fluorophore is quenched when the robot has arrived at the stop location, so decreasing fluorescence indicates completion of the reaction. (a) A diagram of the irreversible reaction on origami. (b) Placing the dock at three different locations. Left, diagrams showing the chosen locations. Right, fluorescence kinetics data of the reaction. The kinetics and completion levels are almost identical among the three, suggesting that the starting location does not affect the reaction. (c) A schematic of the experiment of having one, two, and three dock locations present. A proportional amount of free-floating robot complex was added, so we assume one dock location has one robot, two has two robots, and three has three robots. The fluorescence kinetics to the right shows that the speed and completion level increase with increasing local concentration of robot.

both increase, though the completion level still is not at 100%. This makes sense, as we are effectively doubling and tripling the concentration of robot available to react with the stop. However, the number of stop complexes is limited by the coupling efficiency of inserting complexes into the origami, which in a previous section we determined is maximally about 80%. We also tested adding multiple stop locations with a single dock location. We found that the completion level did not change much with additional stop locations in the absence of additional dock locations, suggesting that it is more likely that the track is missing a robot than that it is missing a stop complex.

Though we were able to achieve some improvement in the reaction rate for the reaction of the robot undergoing four-way strand displacement onto the stop complex, none of the explanations we explored give us insight into the order of magnitude slow down from the expected behavior in some cases, as well as the low completion level. To answer these questions fully will require more experimental investigation. One possibility for the slowdown could be the entropic cost of forming the loop as the robot walks, which could result in the backward reaction being faster than the forward reaction, and perhaps with a longer toehold the geometry is such that this entropic cost is less. We consider this possibility in the next section, as well as discuss additional hypothesis that could explain this behavior.

2.7 Reversible four-way strand displacement on a surface

Investigating reaction rates

We also considered the reversible four-way strand displacement reaction, which is the case when there is a track complex between the robot and the stop complex, as shown in Figure 2.16a. This reaction did not occur in solution. To simulate this system, we used the following set of reactions. $\{4\}$ Rather than being inserted into the origami, in this case the robot is localized onto the origami via the reaction of a $robot : guide_{free}$ complex, with a previously measured reaction rate k_{loc} of $6 * 10^5/M/s$. The robot is now localized onto the track. $\{5\}$ On the $robot : start\&track\&stop_{ori}$ complex, the $robot : track$ complex hybridizes to the track. $\{6\}$ Now $robot : start : track\&stop_{ori}$ can now undergo branch migration. This is a reversible reaction, so there are two rates here, k_{BMf} and k_{BMb}. $\{7\}$ The toeholds dissociate, so now the robot is on the track. The reactions $\{2\}$ and $\{3\}$ from the irreversible case are the last two reactions, in which the robot binds to the stop complex, then undergoes branch migration and dissociation.

Chemical reaction network for a reversible step on origami:

$$\text{robot:guide}_{\text{free}} + \text{start\&track\&stop}_{\text{ori}} \xrightarrow{k_{loc}} \text{robot:start\&track\&stop}_{\text{ori}} \qquad \{4\}$$

$$\text{robot:start\&track\&stop}_{\text{ori}} \underset{k_{\text{d}}}{\overset{k_{\text{h}}}{\rightleftharpoons}} \text{robot:start:track\&stop}_{\text{ori}} \qquad \{5\}$$

$$\text{robot:start:track\&stop}_{\text{ori}} \underset{k_{\text{BMb}}}{\overset{k_{\text{BMf}}}{\rightleftharpoons}} \text{robot:start:track}_{\text{BM}}\text{\&stop}_{\text{ori}} \qquad \{6\}$$

$$\text{robot:start:track}_{\text{BM}}\text{\&stop}_{\text{ori}} \underset{k_{\text{h}}}{\overset{k_{\text{d}}}{\rightleftharpoons}} \text{robot:track\&stop}_{\text{ori}} \qquad \{7\}$$

k_d is the dissociation rate of the two toeholds T_f and T_b. In order to estimate this rate, we began with using Nupack to determine the following free energies. ΔG_J is the free energy of the Holliday junction forming between the robot on its track with the next track via the toeholds T_f and T_b. ΔG_T is the free energy of $T_f + T_b$ binding as a continuous toehold. ΔG_B is the free energy of the branch migration domain. These values for each of the toehold configurations are given in Table 2.3. To calculate the energy penalty, ΔG_P, of forming the 4-way junction, we can use the following formula:

$$L' = L * \frac{\Delta G_T - (\Delta G_T + \Delta G_B - \Delta G_J)}{\Delta G_T}$$

This allows us to calculate the difference in free energy between forming the base pairs in the junction, versus the junction itself. We can then subtract this penalty from ΔG_T to get the free energy of the toehold given this energetic penalty for forming the junction. If we divide by ΔG_T and multiply by L, where L is the length of the continuous toehold $T_f + T_b$, we can estimate L', the length of toehold that the remote toeholds act as. We can then use the formula $k_d = 10^{6-L}/s$. Table 2.4 gives the estimates for L' and the calculated k_d for the three cases we have explored previously. We can also use the local concentrations that we previously calculated to determine k_h, the unimolecular hybridization rate, by the formula $k_h = 2 * 10^6 * C_L$. Those rates are given in Table 2.4 as well.

Toeholds	ΔG_T	ΔG_B	ΔG_J
$T_f = 3, T_b = 3$	-8.76	-64.62	-68.05
$T_f = 3, T_b = 4$	-9.74	-66.05	-70.75
$T_f = 3, T_b = 5$	-11.82	-67.3	-73.13

Table 2.3: ΔG values from Nupack.

With these reaction rates, we can now compare a simulation of the expected system behavior to our data. The forward reaction k_{BMf} should be equivalent to k_{BM}. The backward reaction k_{BMb} should also be equivalent to k_{BM}. In the experimental data, we used $T_f = 3$ and $T_b = 5$. We had three samples, one containing 1 dock location and therefore 1 robot; one containing 2 dock locations and therefore 2 robots; and one containing 3 dock locations and therefore 3 robots.

Toeholds	Calculated L'	Calculated k_d s^{-1}	Calculated k_h s^{-1}
$T_f = 3, T_b = 3$	2.35	$4.47 * 10^3$	$8 * 10^2$
$T_f = 3, T_b = 4$	3.38	$4.18 * 10^2$	$7.24 * 10^2$
$T_f = 3, T_b = 5$	3.95	$1.13 * 10^2$	$6.56 * 10^2$

Table 2.4: Calculated L' and rates k_d and k_h for the reversible reaction on origami.

The idealized simulation is shown in Figure 2.16c, left. The darker trajectories are the fitted k_{BMb} and k_{BMf} for $T_f = 3$ and $T_b = 5$. The lighter trajectories use the k_{BMb} and k_{BMf} fitted for the shorter bridge toeholds, which agreed with each other. Because we would expect these rates to match for all bridge toehold lengths, we show both sets as the range of possible ideal simulation. The simulation was then adjusted to match the data by replacing the idealized k_i with the fitted k_i and tuning k_{BMb} and k_{BMf}. The completion level was also adjusted according to the insertion rate of the guide and stop complexes. The total error for the dock insertion and stop insertion was calculated for each sample, as it was dependent on the number of dock complexes in each sample. The completion level was adjusted for each trajectory according to its specific probability of complete tracks. For example, for the two robot case, the probability was calculated by $(.8 * (1 - (.2^2))$ to get an estimated max completion level of .76. The data and simulation are given in Figure 2.16c, right. The k_{BMb} and k_{BMf} that best fit the data was $2.3s^{-5}$, which is one to two orders of magnitude slower than the fitted branch migration rate for these toehold parameters, depending on which reference k_{BMb} and k_{BMf} are used. If we assume that k_{BMb} is slower than k_{BMf} due to the entropic cost of forming the loop structure, then they must differ by four orders of magnitude to fit the data. This was very surprising, and we explored strategies to build hypotheses for why this slowdown might be happening.

The overall four-way strand displacement reaction is one to two orders of magnitude slower than expected, and the completion level is lower than expected, so we explored some strategies for improving the reaction rate and completion level.

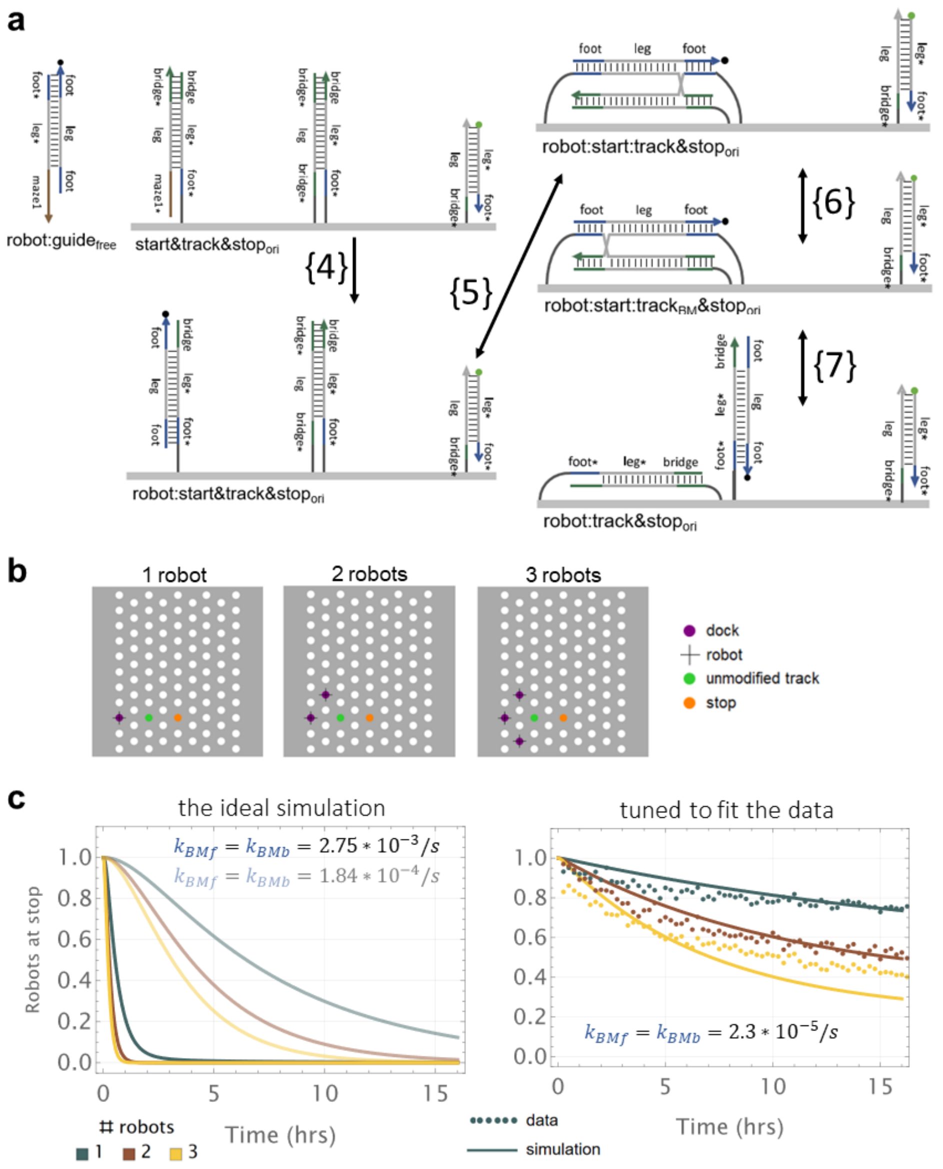


Figure 2.16: Reversible four-strand displacement on origami. (a) Diagrams of the reaction on origami. (b) Diagrams showing the three track configurations, in which there are 1, 2, and 3 robots, respectively. Changing the effective concentration of robots allows us to extrapolate the reaction rates. (c) Left, the idealized simulation for $T_f = 3$ and $T_b = 5$. Right, the fluorescence kinetics data from all three robot configurations with k_{BMb} adjusted for a better fit. The completion level is also adjusted to account for the 80% insertion rate of the dock and stop complexes.

In thinking about speeding up the reaction rate, we determined that the slowest part of the reaction is the branch migration itself. This is true because unlike in three-way strand displacement, in which only one base pair needs to spontaneously pop open in order for the branch migration to proceed, in four-way strand displacement two base pairs on two separate complexes need to simultaneously pop open in order to exchange strands. Thus, destabilizing the helices involved in the reaction without allowing them to dissociate completely should cause the reaction to proceed more quickly, as the base pairs in the helices would breathe more readily. To that end, we explored a mismatch exchange technique. Keeping this in mind, we tried using a mismatch exchange technique to destabilize the track helix and therefore make it easier to invade. We also tried running the strand displacement reaction in sodium buffer rather than magnesium, as in the literature magnesium has been shown to impede the four-way strand displacement reaction (Panyutin and Hsieh, 1994).

Mismatch exchange

We tried one strategy to speed up the branch migration that involves exchanging mismatches between helices. The idea is that each track location has mismatches along the leg domain, which destabilizes the helix such that it should pop open more easily. However, the robot still pairs exactly with the track. As the robot walks, the overall number of paired bases does not change, but it should be easier for the robot to invade along each track location. Similar to in the case of shortening the helix, this strategy is also well-suited to our surface reactions, because in solution, the mismatches might destabilize the helix so much that it dissociates; however, the high local concentration means that the helix would be more likely to stay bound than if it were in solution. The design is shown in Figure 2.17.

We first investigated the stability of helices with 2, 3, and 4 mismatches based on a Nupack analysis. We approximated the tethered track locations as a hairpin for modeling purposes, as that would mimic the unimolecular nature of two track locations held at adjacent locations on the origami.

The mismatch exchange data, shown in Figure 2.18, did not indicate a speedup of the reaction rate with increasing number of mismatches, suggesting that either the branch migration rate is not the rate limiting step or the backwards reaction is still much faster than the forwards reaction.

Monovalent vs. divalent buffers

It has been shown that four-way branch migration proceeds approximately 1000x faster in buffers containing sodium, a monovalent ion, than in magnesium, a divalent ion, though the exact mechanism is unclear (Panyutin and Hsieh, 1994). The standard buffer that we use contains 12.5mM Mg^{2+}, which is divalent.

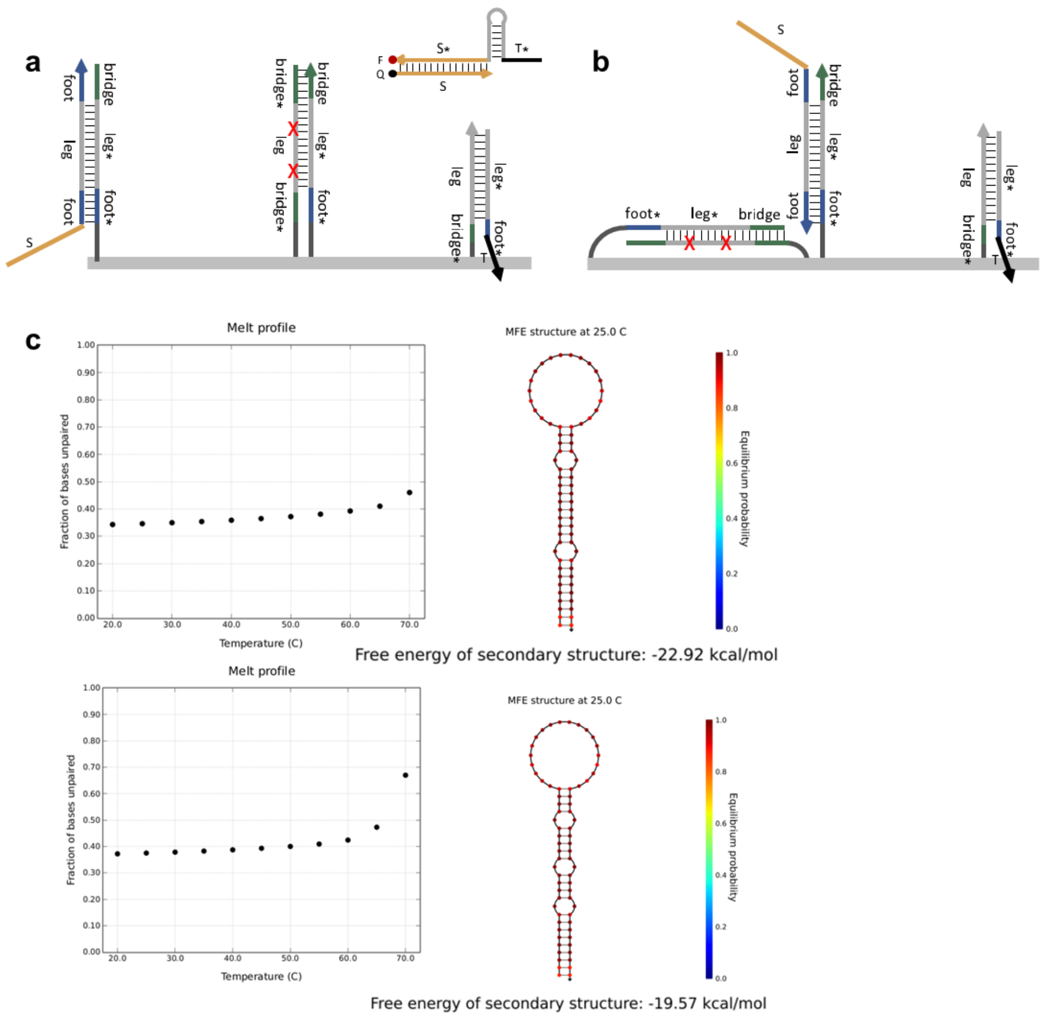


Figure 2.17: Mismatch exchange. (a) A diagram of the starting state in which the robot has been localized to the origami and the track helix has two mismatches. (b) A diagram after the robot has taken a step, in which the mutations are now located in the track loop. The number of base pairs is equivalent between this state and the starting state. (c) Melt profiles from Nupack showing that the track complexes with 2 and 3 mismatches are stably bound at room temperature. We used hairpins to approximate the effect of being tethered at adjacent locations on the origami, as the two strands making up the track helix will bind unimolecularly.

Rather than our standard 12.5mM Mg^{2+} buffer, we used 100mM Na^{+}. We performed a buffer exchange by using 100K filters to purify the origami, washing it with sodium buffer, and then resuspending it in sodium buffer. The data from this experiment showed that there was a slight decrease in the reaction rate, rather than the expected increase.

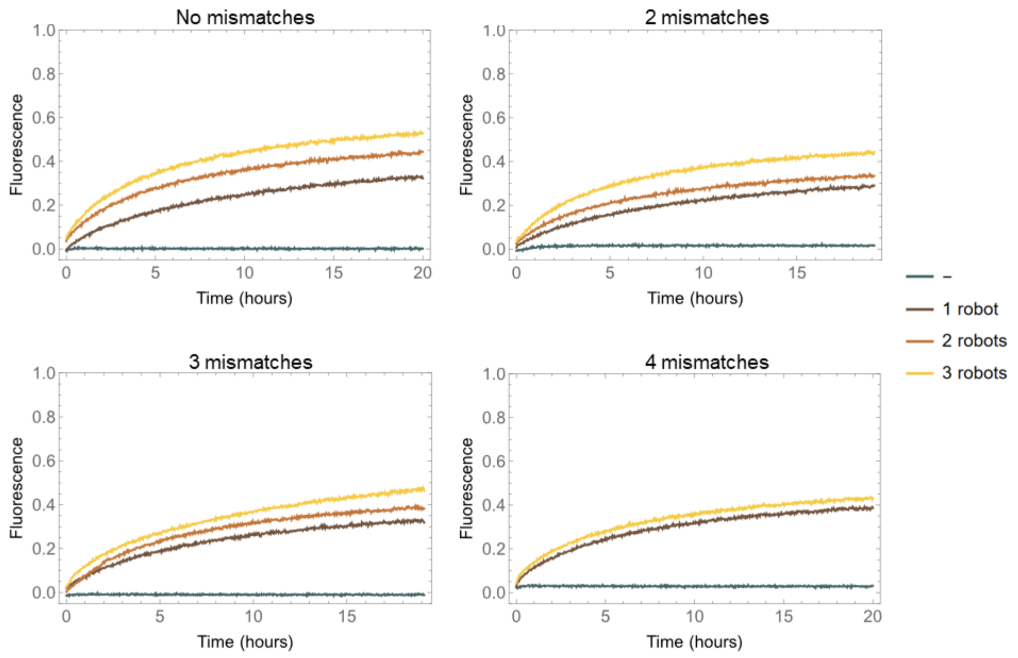


Figure 2.18: Mismatch exchange data shown for four cases, in which 0, 2, 3, or 4 mismatches were introduced.

Probing the entropic cost of forming a loop

Ideally the robot would perform an unbiased random walk on the maze, as this maximizes its likelihood of visiting all track locations in order to find an exit. However, when the robot takes a forward step, the track has fewer degrees of freedom than in its previous state, changing from a double stranded complex attached to the origami by single stranded spacers to a loop, fixed at three points. We see the effect of this entropic loss in the difference between the k_{BMf} and k_{BMb} that we measured. In order to achieve an unbiased random walk, this entropic cost must be compensated for. There are two strategies we explored to achieve this: first, reducing the backward reaction rate by introducing mismatches into the bridge toehold responsible for the backward reaction. Studies have shown that introducing mismatches can significantly slow down strand displacement rates, based on the specific sequence mismatch as well as where the mismatch is placed (Panyutin and Hsieh, 1993; Machinek et al., 2014). However, this strategy is less than ideal because it requires slowing the backward reaction down, rather than speeding the forward reaction up, which is counterproductive to our general goal to build the fastest possible robot. The second strategy we tried was to increase the loop size to relax its geometry, based on the hypothesis that a more constrained geometry has a higher entropic cost of formation.

To test the first strategy, we introduced a mismatch into the bridge* toehold that should slow down the backward reaction rate. Figure 2.19a shows the toehold in which we introduced a mismatch, while b shows that a mutation did not affect the completion level. It also did not affect the reaction rate.

We also tried increasing the length of the single-stranded linker that holds the track strands on the origami. The linkers are included to keep the functional parts of the strands spaced slightly away from the origami, allowing for more geometrical degrees of freedom, which are necessary for any reaction to take place. They also provide a spacer between the strands and the negatively charged origami. The original linker length was a 6T sequence, but in this experiment we decreased the length to 4T to accomodate a 5-nucleotide T_b. A 10T linker length did not seem to make a difference in the speed nor completion level of the reaction, as shown in Figure 2.19d. A 20T linker actually decreases the completion level. We hypothesize that this may be because as the linkers get past a certain length, the system begins to behave more like a solution reaction, rather than a surface reaction. Additionally, the linker strands could still be preventing a reconfiguration step that requires helices to cross over each other.

Here the results of introducing a mutation and the longer linker lengths suggest that the biggest factor slowing down the reaction is not the entropic cost of forming the loop. Instead, we consider other topological, geometrical, and experimental constraints that may be hindering the robot.

Possible constraints of the junction reconfiguration

Our results have indicated that neither the rate of branch migration nor the entropic cost of forming the loop in the forward reaction can fully explain the slow walking behavior of the robot. Here we consider three other possible explanations.

First, we considered whether there might be a topological constraint on the reaction taking place as designed. This could be due to the four-way strand displacement taking place while tethered at three points on the origami. In Figure 2.20a, a diagram of the reaction taking place indicates that there is no topological impossibility. In order to further investigate the effect of having three tether points, we performed an experiment in which one of the strands of the middle track was untethered from the origami, resulting in only two tether points during the four-way strand displacement reaction. This reaction is shown in Figure 2.20b.

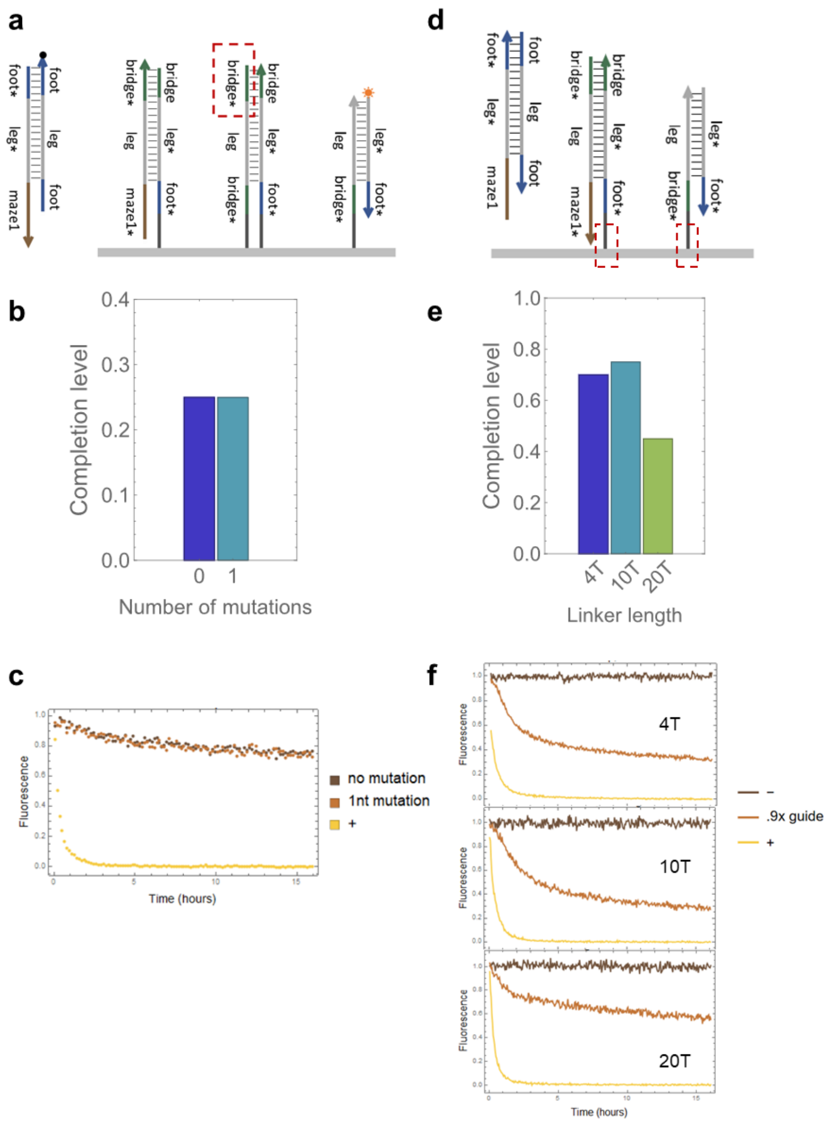


Figure 2.19: Probing the entropic cost of forming a loop. (a) A diagram of the reversible reaction, in which a mutation was introduced into the bridge* domain boxed in red. (b) The completion level of the reaction with no mutations in the bridge* domain and one mutation were the same. (c) Fluorescence kinetics data. (d) A diagram of the irreversible reaction, in which several the linker length was changed. The linker domains of these two track molecules are boxed in red. (e) The completion levels of a 4T and 10T linker were comparable, whereas the completion level decreased once the linker length was increased to 20T. (f) Fluorescence kinetics data.

The tuned k_{BMb}=k_{BMf} for the reaction tethered at two points was $7.36 * 10^{-5}s^{-1}$, which is about a 2.5-fold speedup from the reaction tethered at three points on the origami. However, if we keep k_{BMf} constant and just tune k_{BMb}, now k_{BMb} is four

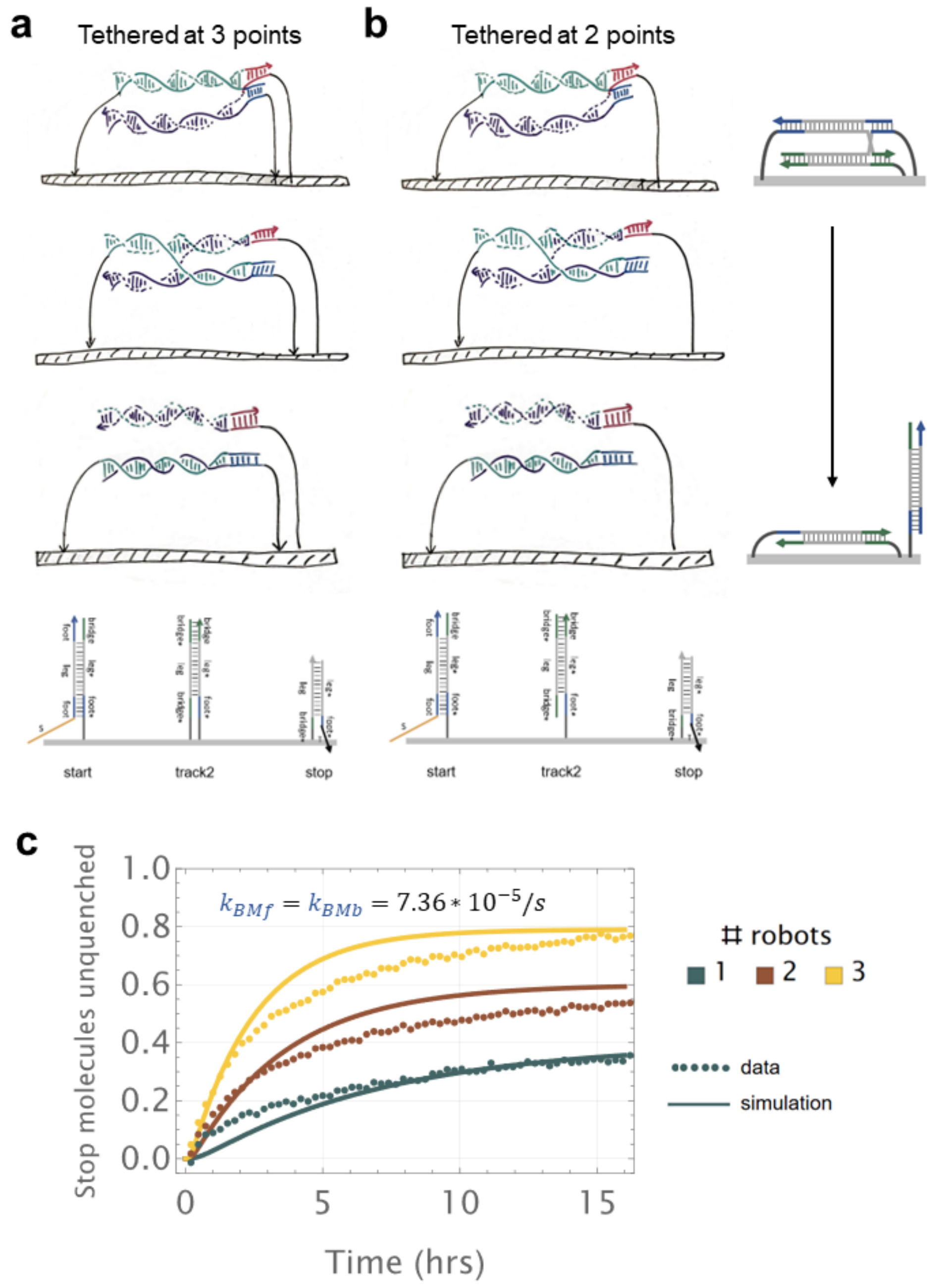


Figure 2.20: Removing one tether point from the reversible reaction. (a) A diagram of the reaction as designed with three tethering points to the origami during four-way strand displacement. (b) A diagram with two tethering points to the origami during four-way strand displacement. (c) Data from the reversible reaction with 2 tethering points, which is slightly faster than the reaction that is tethered at 3 points.

orders of magnitude faster than k_{BMf}, which is a significant improvement. While there did not seem to be topological constraints to the reaction taking place, the three tethering points could be preventing the junction from reconfiguring into a more relaxed open-X form that would allow the reaction to proceed more quickly. While this could explain some of the slow walking behavior of the robot, it did not explain it entirely.

2.8 Discussion

We have identified several design considerations that can be made to improve the functioning of surface-based four-way strand displacement reactions, including increasing the bridge toehold length, building redundancy into the start and stop locations, and using purified strands. We have also shown that removing one tether point of the track increases the overall reaction rate. However, we have been unsuccessful in speeding up the reaction to the expected rate, nor to a rate that would make this robot walking along a longer track feasible.

The discrepancy between our measured rates and the expected walking behavior reveals that there is much we still need to understand about both four-way strand displacement reactions, as well as how they behave on a surface, before we are able to successfully build a track-modifying robot using this mechanism. The surface has potential to interfere with the reaction in two ways: first, by being a large negatively charged platform upon which the reaction takes place, and second, because the track strands are tethered to the surface at multiple points, which reduces the degrees of freedom of the complexes that they are tethering. The negative charge might, for example, result in track locations being trapped on the wrong side of the origami, and unable to pass through it due to repulsion. It might also result in helices being less likely to bend sideways along the origami, which could prevent the helices from reaching the intended reaction.

It has been hypothesized that the conformation of the Holliday junction has a significant effect on the rate of branch migration (Panyutin and Hsieh, 1994). Thus, it seems reasonable to consider that a factor in this case is the number of points at which the four-way strand displacement reaction is fixed to the origami. If the true limitation is the inability of the molecules to appropriately reorganize themselves to accommodate the strand exchange because during the reaction the two helices are fixed at three points, then it may not be possible to adequately speed up the reaction as designed. Four-way strand displacement reactions on a surface may require that only one of the helices involved is fixed, allowing the other helix its full degrees of freedom so the whole structure can isomerize between its stacked structure and a less constrained structure. The speedup in the reaction when one of the middle track locations is untethered indicates that constraining the reaction by three linkers does affect its speed, but it also indicates that there are other factors at play.

Another possibility could be a practical limitation of experimental methods – while strand incorporation into origami has not been directly quantified, we know that about 80% of complexes that we insert after annealing are able to hybridize to the origami. Track strands could be missing or trapped on the wrong side of the origami, and the charge repulsion between the strand and the origami would make it unlikely that the strand would be able to pass through to the correct side. This is particularly likely on a double-layer origami, in which there are two perpendicular layers of helices. We have previously discussed that for this robot design, it is not possible to build redundancy into the track because the robot cannot loop over itself. The best fix for this potential problem given the current design is to build redundancy into the dock and stop locations.

From the perspective of the slow branch migration rate, given that two base pairs need to pop open simultaneously in order for the four-way branch migration to proceed, another route to speeding up the reaction could be to shorten the branch migration domain as much as possible, while still considering that the length of the leg domain should span an integer number of turns. This strategy is particularly well-suited for surface reactions, because a double-stranded domain that may not be stable in solution may be stable when both strands are tethered in close proximity to each other, as this increases the local concentration of the strands and causes them to behave more like a hairpin, unimolecularly. Thus, a shorter helix could be more stable on origami than in solution. For the start and stop locations, in which one strand is not fixed to the origami, it may be possible to add additional anchoring domains that provides the extra stability needed when both strands are not bound to the origami.

Though we have explored many strategies for speeding up the irreversible and reversible four-way strand displacement reactions on a surface, there are still many questions remaining surrounding the factors that affect the reaction speed and completion level. We hope that this exploration in pursuit of a maze-solving DNA robot will help guide the further investigation and understanding of these kinds of reactions, as well as provide the basis for building this robot in the future.

2.9 Future Experiments

The model used a theoretical estimate of k_d, and from the model we were able to determine k_{BM}. Both of these parameters could be determined experimentally by using the right experiment, which would validate the theoretical estimates that were made for the rates in the model as well as the rate that the model was used to find.

The irreversible reaction in solution can be used to find k_{BM}. In the model from Dabby, 2013, k_1 is the rate of successful initiation of four-way strand displacement and k_2 is the rate of four-way branch migration. k_2 is equivalent to k_{BM}. If the bridge toehold, T_b is extended (>10nt) such that k_1»k_2, the rate of the reaction will be equivalent to k_2. Another way to validate the same rate parameter is to keep T_b at 3, 4, and 5 nt, but increase the concentrations of the complexes as much as possible.

To find k_d, the reversible reaction in solution can be used. This reaction would have to be performed at high concentration, as at low concentration it does not take place. Given the k_{BM} found in the experiment outlined above, the remaining unknown in a solution experiment is k_d of the two sets of toeholds.

Finally, it would be useful to repeat the irreversible four-way strand displacement on origami to determine whether the k_{BM} is accurate, given that it is the same for T_b = 3nt or 4nt, but faster for T_b = 5nt. It should theoretically be the same in all three cases.

C h a p t e r 3

SHAPE RECONFIGURATION IN DNA TILE ASSEMBLIES

3.1 Introduction

DNA origami is a powerful technique for self-assembling nanoscale structures with arbitrary 2D and 3D shapes. It is what is commonly referred to as a "bottom-up" nanofabrication approach, in contrast to a "top-down" approach like lithography. Lithography is responsible for the production of transistors with nanoscale feature sizes that have powered the electronic revolution, but it is very expensive. DNA origami, on the other hand, is inexpensive and scalable. The parallelism of DNA can be leveraged to simultaneously self-assemble millions of copies of the designed system at the same time.

While there have been demonstrations of interesting 2D and 3D shapes designed using the principles of DNA origami, here we will focus on tiles, which are flat with four straight edges. While the mechanism of DNA origami, particularly the use of short staple sequences, makes any shape addressable, tiles are particularly useful because their flat surfaces can be used to spatially organize other molecules or materials according to specific patterns. They can be used to organize, for example, nanoscale devices, biochemical circuits and molecular robots (Kopperger et al., 2018; Chatterjee et al., 2017; Thubagere et al., 2017). The edges of DNA origami tiles can also be engineered to interact, so they can form supramolecular structures called arrays or assemblies. Tile assemblies are engineered via sticky edge interactions and may be finite or unbounded. This is not limited to square tiles, though more straightforward with regular edges that allow straightforward tessellation. Tiles can be designed to interact in such a way that they create arbitrary patterns with programmed global properties, or they can be designed to create finite arrays with fixed patterns (Tikhomirov, Petersen, and Qian, 2017b; Tikhomirov, Petersen, and Qian, 2018). Some demonstrations of the power of DNA origami tiles are summarized in Figure 3.1.

By definition, finite arrays form a static assembly. Until recently, they have been unable to change shape once formed - that is, they were designed to achieve a certain shape, and once that shape was formed, they could not be further reconfigured. The only programs they were capable of executing were binding and unbinding. Similar

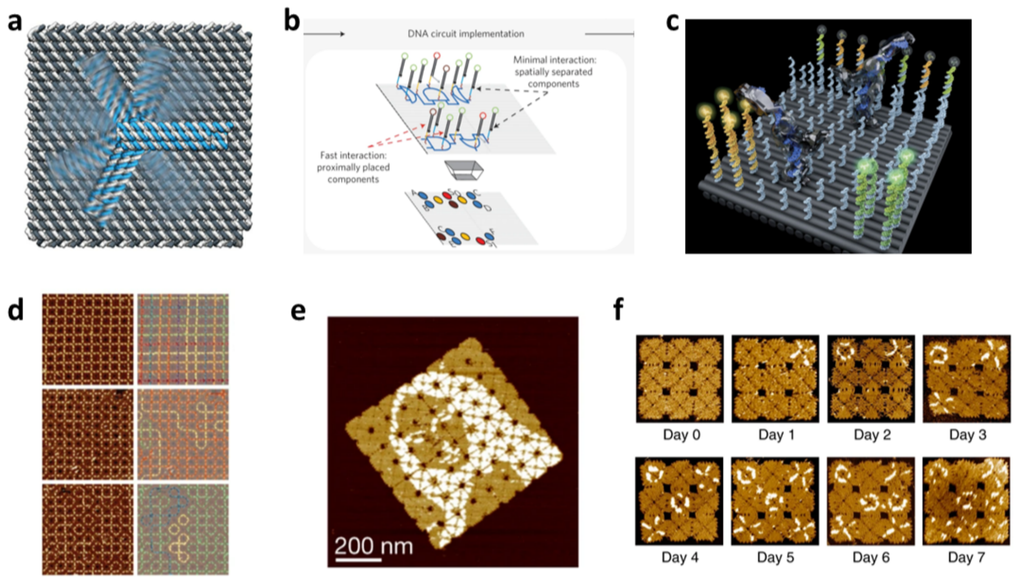
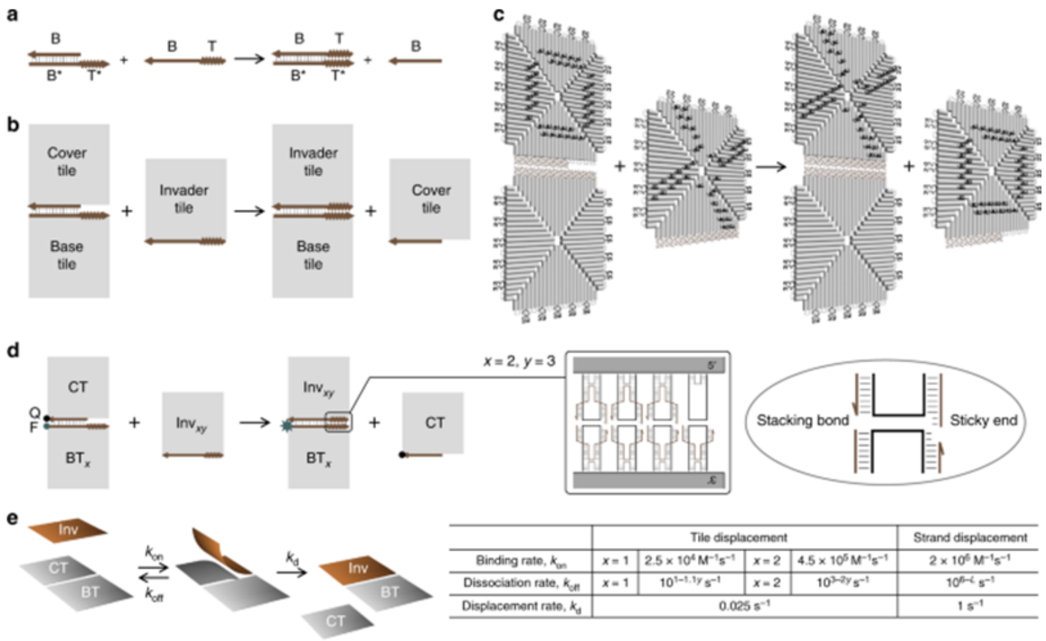

a
b
DNA circuit implementation
Minimal interaction: spatially separated components
Fast interaction: proximally placed components
c
d
e
200 nm
f
Day 0
Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7

Figure 3.1: Examples of systems that use DNA origami. (a) A molecular robotic arm that is actuated by electrical fields. Adapted from Kopperger et al., 2018. (b) A DNA circuit that is organized on a DNA origami. Adapted from Chatterjee et al., 2017. (c) A cargo-sorting robot that walks on a DNA origami track. Adapted from Thubagere et al., 2017. (d) Programmed disorder in infinite arrays. Adapted from Tikhomirov, Petersen, and Qian, 2017b. (e) A Mona Lisa made by hierarchically assembling 64 DNA tiles with patterns on them. Adapted from Tikhomirov, Petersen, and Qian, 2018. (f) A tic-tac-toe game played using swap reactions of DNA origami tiles, called tile displacement. Adapted from Petersen, Tikhomirov, and Qian, 2018.

to how seesaw gates interact to rearranges DNA complexes according to an input signal, it would be powerful to be able to rearrange an assembly that has already been formed. For example, the information encoded into the tile interactions could direct the functioning of a device, circuit, or robot that is localized on the origami by reorganizing its environment.

Petersen, Tikhomirov, and Qian, 2018 showed that tile edges can be engineered to facilitate a technique called tile displacement in which origami tile assemblies are reconfigured by an invader array(s), as shown in Figure 3.2. These origami tiles are designed to have edge staples along each side. While in strand displacement the toehold is a continuous single-stranded sequence, in tile displacement the toehold is a 2-nucleotide truncation in a series of edge staples, which exposes a 2-nucleotide region of the scaffold loop. The invader has 2-nucleotide extensions on its edges that match the exposed scaffold loops, and can therefore bind at those spots. The branch migration domain is the remaining edges, bound by the same 2-nucleotide extensions. This reaction allows for tile-based programs that can now execute a swap reaction. Petersen, Tikhomirov, and Qian, 2018 demonstrated the technique on 4 and 9-tile arrays with single tile or dimer invaders. One example, of a tic-tac-toe game played via these kinds of swap reactions, is shown in 3.1f.



	Tile displacement				Strand displacement
Binding rate, k_{on}	$x = 1$	$2.5 \times 10^4\ M^{-1}s^{-1}$	$x = 2$	$4.5 \times 10^5\ M^{-1}s^{-1}$	$2 \times 10^6\ M^{-1}s^{-1}$
Dissociation rate, k_{off}	$x = 1$	$10^{1-1.1y}\ s^{-1}$	$x = 2$	$10^{3-2y}\ s^{-1}$	$10^{6-L}\ s^{-1}$
Displacement rate, k_d	$0.025\ s^{-1}$				$1\ s^{-1}$

Figure 3.2: Concept and kinetics of DNA tile displacement. (a) Domain-level diagram of a DNA strand displacement reaction. T is a short toehold domain of typically 3 to 8 nucleotides. B is a long branch migration domain of typically 15 to 20 nucleotides. Asterisks in the domain names indicate sequence complementarity. (b) Domain-level and (c) origami-level diagram of a DNA tile displacement reaction. Toehold and branch migration domains are composed of 4 and 7 edge staples, respectively. (d) Tile displacement reactions. CT is a cover tile. BT_x is a base tile with x=1 or 2 indicating the number of nucleotides in the sticky end of each of the 4 edge staples in the toehold domain. Inv_{xy} is an invader tile with y=0 through 4 indicating the number of edge staples in the toehold domain. F and Q indicate a fluorophore- and quencher-labeled edge staple, respectively. (e) Model and rate parameters of tile displacement in comparison with strand displacement. L is the number of nucleotides in the toehold domain of a strand displacement reaction with average DNA sequences. Adapted from Petersen, Tikhomirov, and Qian, 2018.

While tile displacement has been shown to be a powerful mechanism for reconfiguring origami arrays, it was not yet known how robust the reaction was as parameters changed, such as edge configurations and identities. It was also not known how complex of arrays or assemblies could be used as an invader. This has implications for what kind of information can be encoded in these swap reactions.

In this work, we aimed to expand tile displacement beyond simple binding, unbinding, and swap behaviors in arrays and show that it can be performed by arbitrarily shaped tile assemblies. We also aimed to further characterize the design space for these kinds of reactions. Demonstrations of more complex shape reconfigurations would push the theoretical limits of what kinds of computations could be achieved with tile displacement. This could inform the possibility of experimentally implementing tile displacement reactions that perform complex computations via the information encoded in the tile interactions.

3.2 System design

The shapes of the original assembly and invading assemblies were designed by a student in the BE/CS 196A class, The Design and Construction of Programmable Molecular Systems, according to a specific set of guidelines including the number of tile types involved. We then corrected the edge design to be compatible with previously established principles of tile displacement.

The system is comprised of an initial tile assembly, a sword, and two invaders, the snake head and the snake tail. The sword contains 7 tiles, with one tile repeated 4 times. The head is made up of 4 unique tiles. The tail is comprised of 3 tiles, though it is designed to grow arbitrarily long polymers. The invaders reconfigure the sword into a snake via toeholds along the yellow and brown edges of the middle tile in the sword, producing two waste products, the sword blade and the sword handle. A diagram of the full system is shown in Figure 3.3.

The tile displacement toeholds are located along interacting edges, in which the edge staples on one side have 2-nucleotide truncations that reveal 2 nucleotides of the scaffold loop, which we call a receiving edge. On the other tile, the edge staples have 2-nucleotide extensions that are complementary to the exposed scaffold loops, which we call the giving edge. To create the toehold, the edge staples are left out altogether, which allows an invading edge that contains all of its edge staples with 2-nucleotide extensions to bind to the exposed scaffold loops and initiate branch migration. It is desirable to have short sticky ends so that there is enough breathing to allow the invading tile to branch migrate. Inert edges, which do not interact with other edges, are made up of edge staples that contain hairpins at their ends. Hairpins are necessary to prevent stacking interactions from forming at the ends of the edge staples that would cause tiles to stick together.

Each edge contains 11 edge staples. We can use an edge code to define the compo-

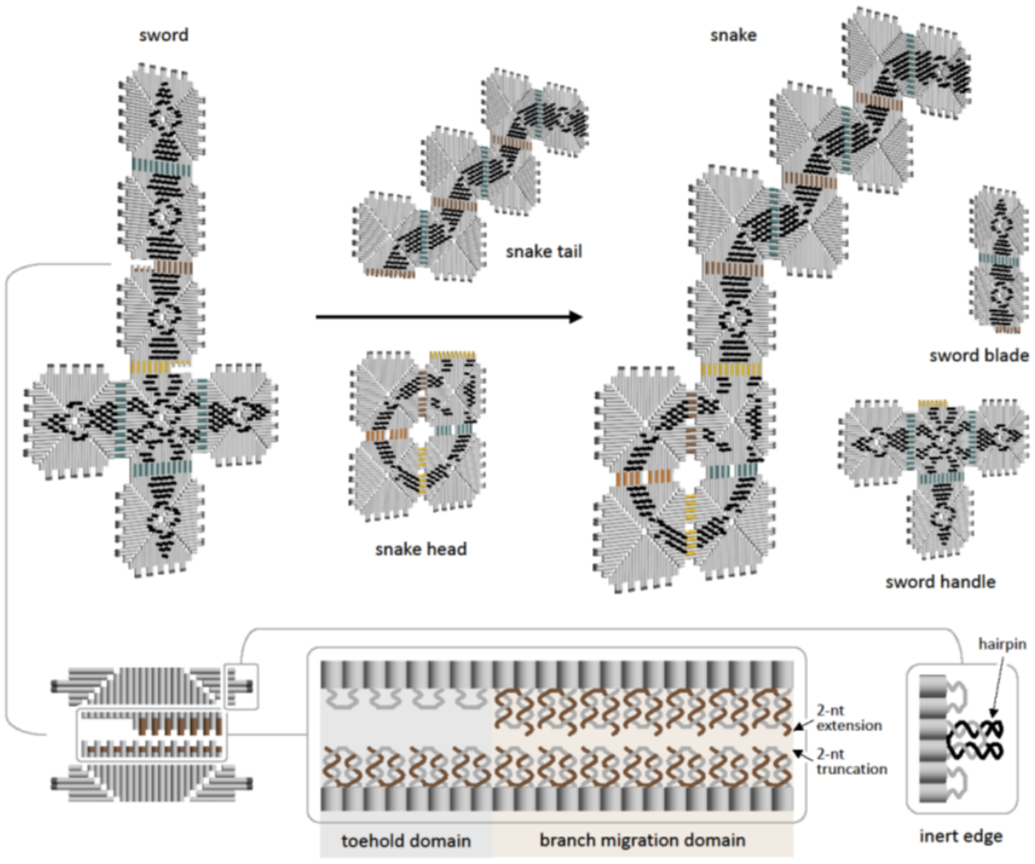


Figure 3.3: Schematic of the tile displacement assembly system. The sword is made up of 7 tiles, with two tile displacement toeholds surrounding the middle tile. It is reconfigured by the snake head and snake tail into a snake, which results in two waste products, the blade and handle. Edge interactions between two tiles are via a giving edge, which has 2-nucleotide extensions on its edge staples, and a receiving edge, which has 2-nucleotide truncations on its edge staples. The toehold is 3 to 4 edge staples long, while the branch migration domain is 7 to 8 edge staples long. Inert edges are created by hairpins at the ends of the edge staples.

sition of those edges, in which the first number tells us the configuration of the first edge staple on a particular edge, counting clockwise from the left. For example, 22222222220 would indicate an edge that has 2-nucleotide extensions on its first 10 edges staples, then a stacking bond on its final edge staple. A # is used to denote an inert hairpin edge. A _ is used to denote the absence of an edge staple, leaving just a scaffold loop.

3.3 Experimental implementation

The origami tile used here is a single-layer tile, unlike the double-layer tile used for the robot. In this case, we want the flexibility of the single-layer origami. It is made up of four triangles that are identical in structure, though not in sequence. They are held together at the seams by single-stranded bridge staples, which allows for flexibility along the seams. Bending along the seams between the triangles that make up the tile is likely what allows one tile of the original array to move out of the way when the invader binds to the toehold. A secondary consideration is that we are limited by the size of the M13mp18 scaffold, a single-layer tile has a larger area, and therefore more edge staples, than a double-layer tile.

A schematic of a representative tile is shown in Figure 3.4, though it is smaller than the one we use. This tile shows 6 edges, while the tiles that we use have 11. The four triangles, which are all the same size but have unique fixed sequences dependent upon the sequence of the M13mp18 scaffold, are called T1, T2, T3, and T4. We code giving edges as G1, G2, G3, and G4, dependent on the triangle to which they are giving. A giving edge is defined by the code G_xT_y, where $x, y \in \{1, 2, 3, 4\}$, defining the identities of the 2-nucleotide sticky ends and then the edge staples they are extended from. Receiving edges are coded R1, R2, R3, and R4.

Each tile is annealed in a separate tube using a standard 6 sec/.1°C protocol with 5x excess staples. Most of the extra staples in solution should not affect the tile displacement reaction, with the exception of the edge staples. The scaffold loop sequences at each edge are identical across all tiles, so if we were to mix the tiles together after annealing, the presence of different edge staples for the same edges could result in the incorrect edge staple hybridizing to or invading at that edge on a different tile. To avoid this problem, we add 25x of what we call negation strands after annealing each tile – these are complementary to all edge staples, so any edges not incorporated into the tile in question will be essentially turned into waste by the appropriate negation strand. The staple concentration is at 5x, so we add them at 25x so that they are at 5x relative to the staple concentration.

All subsequent anneals begin at 50°C, which is below the melting temperature of DNA origami. If we were to begin annealing at a higher temperature, we would risk melting the tiles themselves. Though the negation strands should hybridize quickly at room temperature to complementary single strands, we do a quick cool down from 50°C to 20°C at 2 sec/.1°C to ensure maximal binding of excess edge staples. Finally, the tiles are mixed together in the appropriate ratios (e.g., the sword

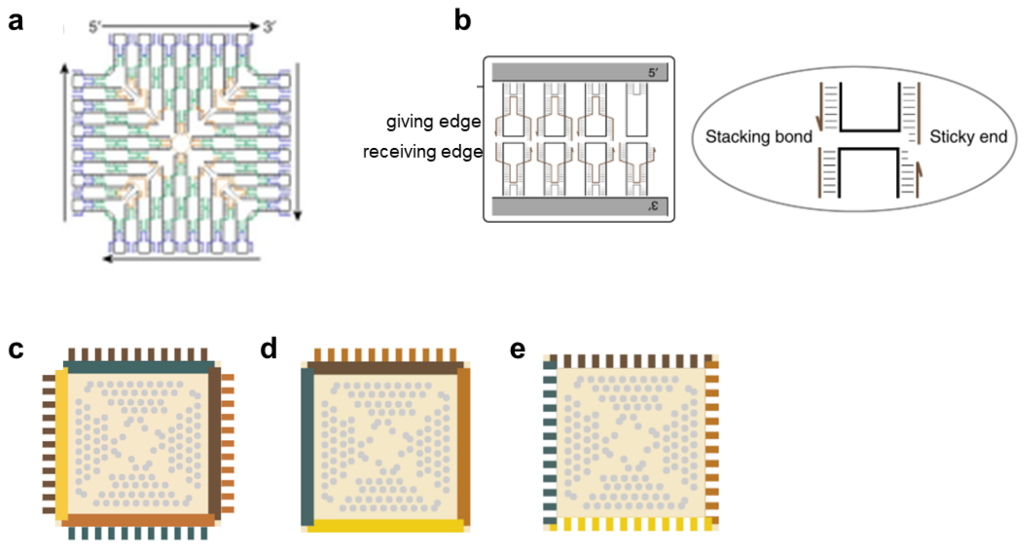


Figure 3.4: An example tile used for tile displacement. (a) A representative tile, made up of four mirrored triangles held together by bridge staples. This tile has six edges, while the tiles we use have eleven edges. Blue lines represent edge staples, green lines represent internal staples, and orange lines represent bridge staples. (b) A diagram of a giving edge on one tile, which has 2-nucleotide extensions on its edge staples, and a receiving edge on another tile, which has 2-nucleotide truncations on its edge staples. This configuration allows the edge staple of a giving tile to bind to the exposed scaffold loop of the receiving tile on one side of the edge and stack on the other side. Adapted from (Petersen, Tikhomirov, and Qian, 2018). (c) An abstraction that we use to represent tile systems. The blue edge is T1, the brown edge is T2, the orange edge is T3, and the yellow edge is T4. In this case, the edges clockwise from top are $G2T1$, $G3T2$, $G1T3$, $G2T4$. The gray dots on the tile represent staple junctions where an extension could be added to create a pattern. (d) A tile with one edge $G3T2$, and the rest inert edges. (e) A tile with all receiving edges.

would be 1:1:1:4, as there are four of the same tile in the assembly). They are then annealed from 50°C to 20°C at 2 min/.1°C.

To create patterns on the tiles, the staples at those locations are extended by a 20T sequence, or 20 consecutive thymine nucleotides. 20T is the chosen sequence because it is unlikely that it would spuriously interact with any active edge sequences, given that the A-T bond is weaker than the C-G bond. A single strand of DNA is very flexible, and therefore when it is imaged by an AFM it is easily pushed out of the way by the tip. It is very difficult to visualize a single strand of DNA on origami. Therefore, directly before imaging a complementary 20A sequence is added to the sample. 20A hybridizes to 20T to form a rigid double helix that cannot be pushed out of the way by the tip. The height difference between the helix extension and the surface of the tile is enough that the helices show up in contrast to the tile, providing the patterns that we see.

3.4 Establishing design principles

In order for tile displacement to be a useful tool, it is important to understand how robust it is to various characteristics of the involved edges. To better characterize the reaction rates of tile displacement, we investigated a variety of toehold and branch migration domain configurations. We performed the simple dimer experiment shown in Figure 3.2d. The edges that we evaluated had 3-staple toeholds and 8-staple branch migration domains, one of which was a fluorophore-quencher pair, which has been shown to have binding energy similar to a 2-nucleotide sticky end.

We evaluated four configurations of toehold and branch migration domain. In two the cases, the toehold and the branch migration domains were the same type of edge, e.g. both giving edges or both receiving edges. In the other two cases, the toehold and branch migration were opposite types of edges. In one case, the toehold was a giving edge and the branch migration domain receiving, and in the last case vice versa. These configurations are shown in Figure 3.5a.

In order to estimate reaction rates from our data, we used the model from Petersen, Tikhomirov, and Qian, 2018. The set of reactions describing the tile displacement reaction are:

$$\mathrm{CT:BT:Inv} \underset{10^{3-2*y}/M/s}{\overset{k_{on}}{\rightleftarrows}} \mathrm{CT:BT:Inv} \tag{8}$$

$$\mathrm{CT:BT:Inv} \xrightarrow{.025\ s^{-1}} \mathrm{CT + BT:Inv} \tag{9}$$

where y refers to the number of edges in the tile displacement toehold. In the Petersen, Tikhomirov, and Qian, 2018 paper, the maximum k_{on} was reported to be $4.5 * 10^5 M^{-1} s^{-1}$.

We also investigated the effect of a discontinuous toehold, in which the edges in the toehold are non-adjacent. Though all edges in a toehold are effectively remote toeholds from a strand displacement perspective, it was unclear whether the already weak toehold edge interactions, once spaced farther apart from each other, would be strong enough for branch migration to be initiated. The five toehold and branch migration configurations that we tested are shown in Figure 3.5.

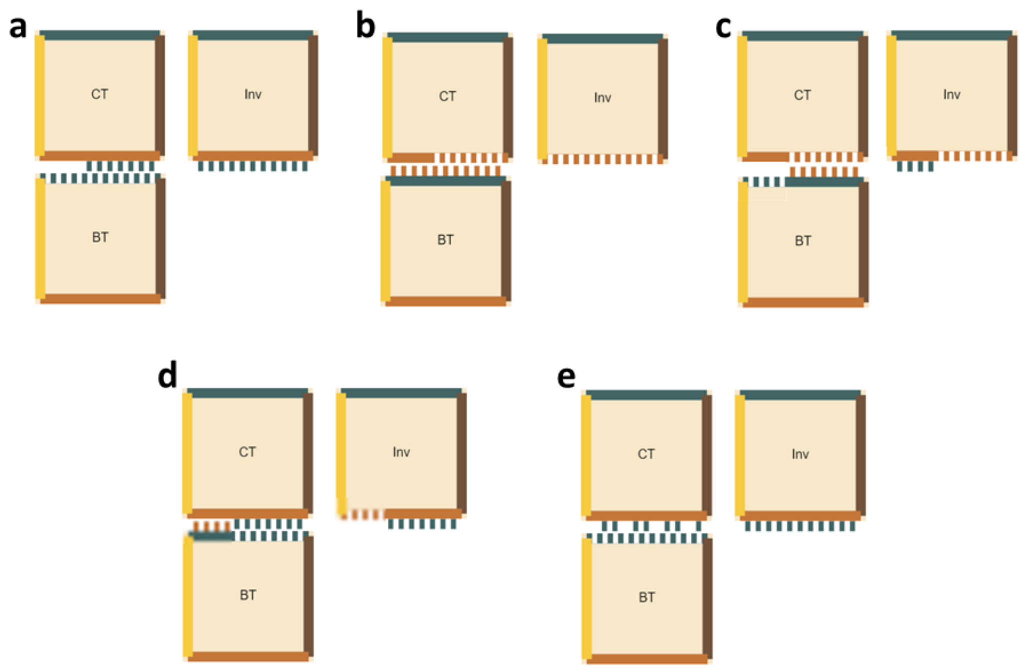


Figure 3.5: Tested edge configurations. The schematics shown are examples of these tile configurations, but not the ones actually tested. Refer to Table 3.1 for edge identities used in experiments. (a) A continuous giving edge toehold and giving edge branch migration domain. (b) A continuous receiving edge toehold and receiving edge branch migration domain. (c) A continuous giving edge toehold and receiving edge branch migration domain. (d) A continuous receiving edge toehold and giving edge branch migration domain. (e) A discontinuous giving edge toehold and receiving edge branch migration domain.

The rates of tile displacement by the invaders tested were similar, as shown in Table 3.1. This suggests that any edge configuration can be used interchangeably when designing tile displacement systems. This was also an interesting exploration because it opened up the possibility of having tile displacement edges with the same toehold and orthogonal branch migration domains, so that they are not just orthogonal by sequence, but also by type of edge. This could increase the design space for tile displacement systems with many reconfiguration steps, though there would be the trade-off of some toehold occlusion by the wrong tile if the same toehold were to be used in two places. Discontinuous toeholds could also be useful, for example to design two orthogonal tile displacement reactions that use the same receiving edge base tile.

Continuity	Toehold	Branch migration	$k_{on}\ M^{-1}s^{-1}$
continuous	$G2T3$	$G2T3$	$8 * 10^4$
continuous	$R2$	$R2$	$1 * 10^5$
continuous	$G3T2$	$R2$	$2 * 10^5$
continuous	$R3$	$G2T3$	$5 * 10^4$
discontinuous	$G2T3$	$G2T3$	$1 * 10^5$

Table 3.1: Reaction rates from various edge configurations.

As noted previously, giving edges were designed to be complementary to the exposed scaffold loop of the receiving edge corresponding to that triangle. The sequence of the scaffold loop was predetermined by the M13mp18 sequence, and was static for the four triangles that we call T1, T2, T3, and T4. We discovered that when a triangle was giving to itself, or $G_xT_x \in 1, 2, 3, 4$ – e.g., T4 is giving to T4 – there was a significant slowdown in the rate of the tile displacement reaction. The hypothesis for why this was the case became clear upon comparing the interactions between two invaders that are giving to a different triangle than themselves G_xT_y, as in Figure 3.6a, to two invaders that are giving to the same triangle as themselvesG_xT_x, as in Figure 3.6b.

In the G_xT_y case, the only sequence overlap between an invader and itself was by random chance, with some specific examples given by the green arrows and underlined sequences. On the other hand, in the G_xT_x case, there was a 2-nucleotide sequence overlap at every sticky end and stacking bond. This was because the sticky ends were designed to bind to the scaffold loop revealed by a 2-nucleotide truncation at that same location on the sequence. So this could only occur if the giving edge were designed to give to its same triangle. Because there were 22 locations where a 2-nucleotide branch migration could take place, this invader could sequester itself via these transient interactions and thus would be less available to undergo the tile displacement reaction. The k_{on} for a continuous, giving edge toehold and giving edge branch migration domain G4T4 invader was $3 * 10^3 M^{-1}s^{-1}$, which was an order of magnitude slower than what had been measured in cases where the triangle and giving edge did not match, as shown in Table 3.2. The kinetics for all rates are shown in Figure 3.7. In future sections, we explore the effects of redesigning the tile interactions within the arrays such that there are no G_xT_x edges.

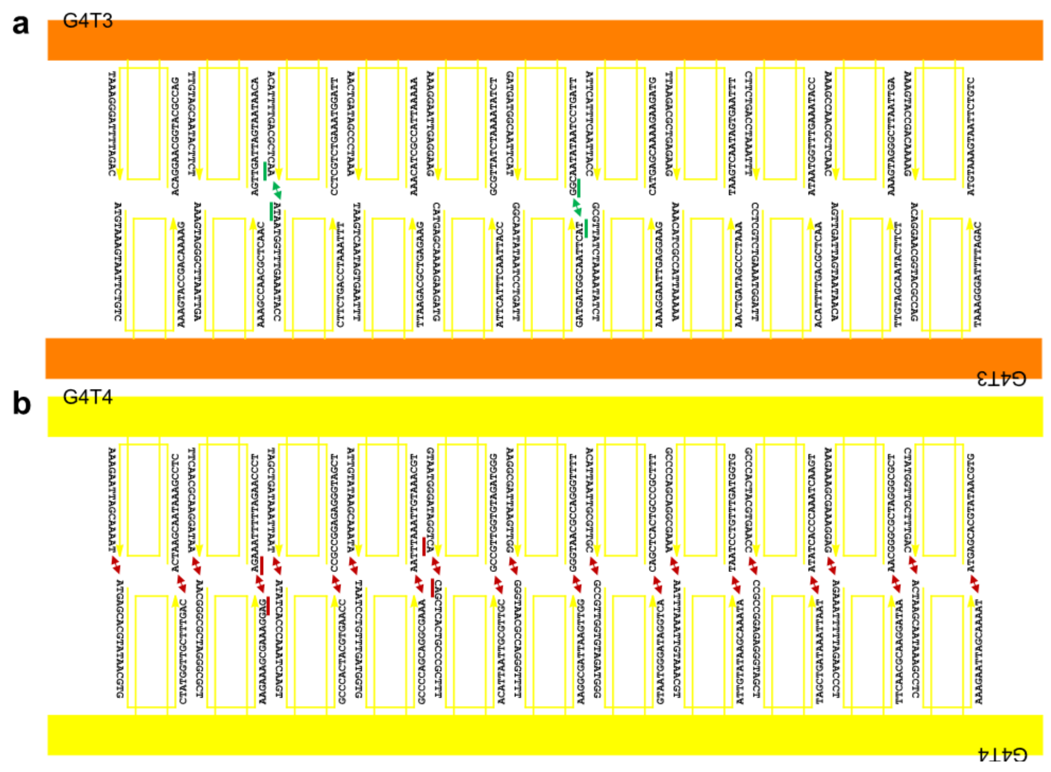


Figure 3.6: Spurious invader interactions. (a) An edge G4T3 shown interacting with itself, where the orange bar represents the tile and the yellow lines indicate edge staples. In this case, the only sequence overlap between the sticky ends and stacking ends of the invaders is by random chance. A couple of examples are given by the green arrows and underlined sequences. (b) An edge G4T4 shown interacting with itself, where the yellow bar represents the tile and the yellow lines indicate edge staples. In this case, at every sticky edge and stacking end of the edge staples, where the red arrows are, there is a 2-nucleotide sequence overlap. A couple of examples are underlined in red. This means that a 2-nucleotide branch migration can occur at each of those 22 locations.

Configuration	Toehold	Branch migration	Rate $M^{-1}s^{-1}$
$GxTy$	giving	giving	$8 * 10^4$
$GxTx$	giving	giving	$3 * 10^3$

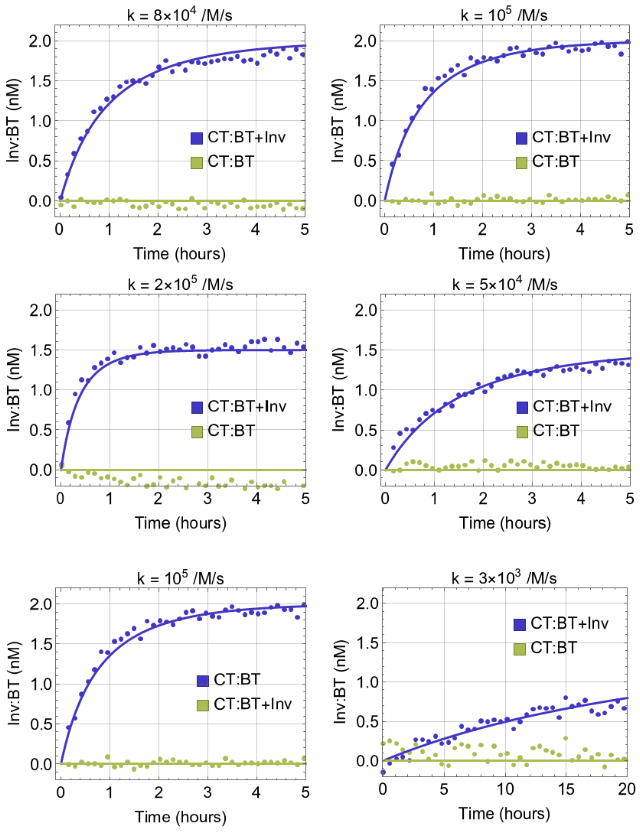
Table 3.2: Comparing the reaction rates for $GxTy$ and $GxTx$.

3.5 Improving array formation

We estimate yield in array formation via large-scale AFM images that are 5-10 microns in height and width. The formula we use for this estimation is:

$$CT\!:\!BT + Inv \xrightarrow{k} Inv\!:\!BT + CT$$

a



b

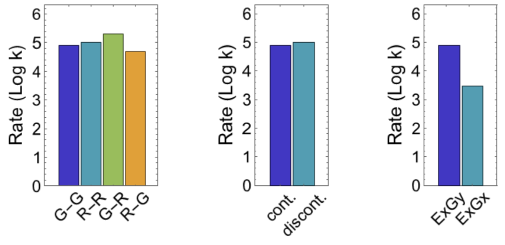


Figure 3.7: Kinetics of dimer reactions. (a) Fluorescence kinetics data for the first five G_xT_y invader configurations, and bottom right for the G_xT_x configuration. (b) Charts summarizing the rate comparison of different edge configurations.

$$\frac{a_{correct}}{a_{correct} + a_{incorrect}}$$

where $a_{correct}$ refers to the total area of correctly formed structures and $a_{incorrect}$ refers to the total area of incorrectly formed structures.

A complete structure of the sword array is shown in Figure 3.8a. The initial yield estimate of the sword structure was 38.2%. An analysis of the kinds of spurious structures that are forming can provide insight into the mechanistic problem preventing the array from forming correctly. We noticed many one-armed swords, one of which is shown in Figure 3.8b. We posited that this may be due to the formation of a 3D structure in solution that would prevent one of the sword arms from attaching in its correct location, but that we would not see because the AFM imaging technique flattens structures on the imaging surface. To test this hypothesis, we tried annealing the sword in lower salt concentrations than the standard 1x=12.5mM Mg^{2+}, which has been shown to reduce the formation of 3D structures (Tikhomirov, Petersen, and Qian, 2018). We tried .5x and .25x salt concentration. While the .25x salt concentration did result in a higher yield than the .5x, as shown in Figure 3.9a, they were both lower than the original.

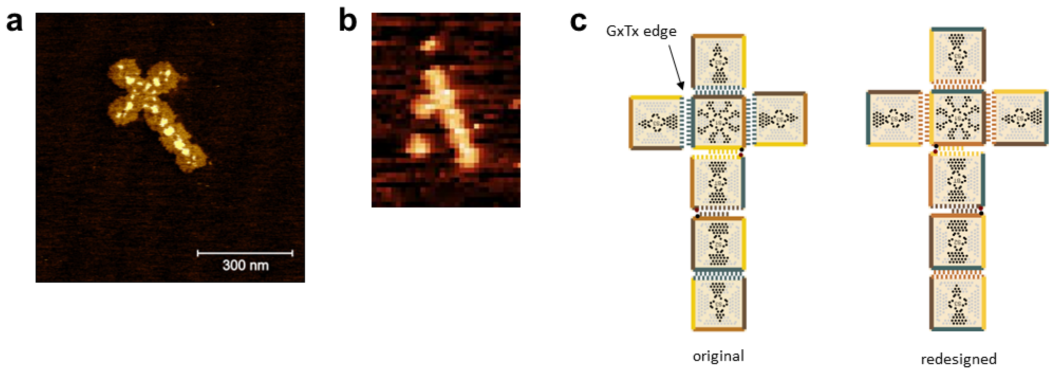


Figure 3.8: The sword array. (a) An AFM image of the sword array showing the details of its pattern. (b) A one-armed sword, one of the most common spurious structures observed in the original sword. (c) Diagrams of the original sword design, left, and the redesigned sword to avoid G_xT_x edges.

A longer annealing time could also improve the yield of the assembly by allowing more time at the critical temperature for the structures to rearrange into their lowest energy state. We tried an annealing protocol of 5min/.1°C, which resulted in a yield of 25.5%. The yield analysis is shown in Figure 3.9b.

We also tried a hierarchical assembly approach. It has been shown that annealing a larger array in stages can decrease the spurious interactions at any given stage in the assembly process, resulting in a higher yield of correctly assembled arrays (Tikhomirov, Petersen, and Qian, 2017b). The three approaches that we took are shown in Figure 3.9c, d, and e. None of these three approaches resulted in a higher yield than the original, non-hierarchically assembled sword.

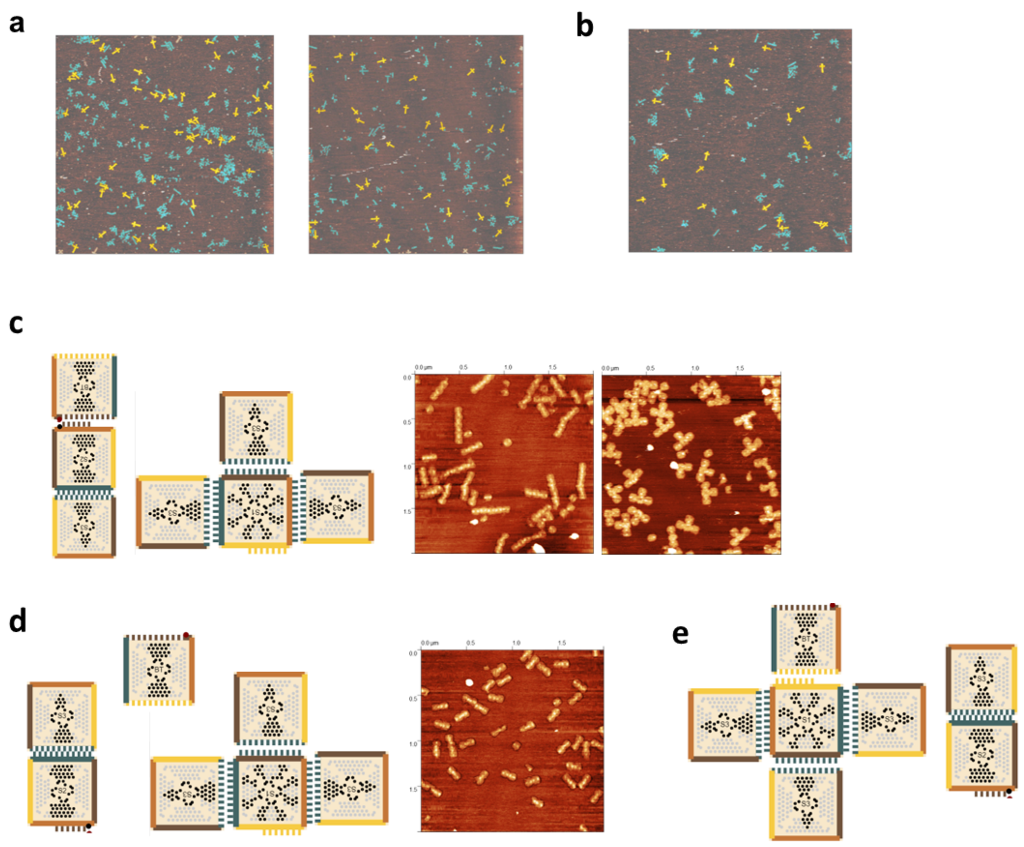


Figure 3.9: Techniques for improving the yield of the sword array. (a) Yield analysis for .5x Mg^{2+}, left, 25.9%, and .25x Mg^{2+}, right, 29.7%, where 1x=12.5mM Mg^{2+}. (b) The yield analysis from a sword that underwent a longer annealing protocol from 50°C to 20°C at 5min/.1°C. The yield was 25.5%, lower than the original. (c) The first of three hierarchical assembly approaches, using two intermediate arrays, with example AFM images of the intermediate structures shown. For each, the intermediate arrays were annealed from 50°C to 20°C at 2min/.1°C, then mixed together and annealed from 30°C or 40°C to 20°C at 2min/.1°C. In this case, the 3-tile array formed long rods, suggesting that there is a spurious interaction between the exposed active edge and itself. It had a 33% yield. The four-tile structure had an 82% yield. The yield of the full sword in the 30°C and 40°C case were both about 22%. (d) The second approach with an example AFM image of the 2-tile intermediate structure, which formed with a yield of 96%. The best yield of the full sword was the 40°C case, at 30%. (e) The third approach, which had a yield of 28% in the 30°C case and 35.6% in the 40°C case.

We also noticed that in the original sword design, one of the sword arms attached at a G1T1 edge, as shown by the blue edge giving to the blue receiving edge in the left diagram in Figure 3.8c. One of the tile displacement edges was also a G4T4 edge. As we had learned that this kind of edge is likely to spuriously interact with itself, we redesigned the sword such that there were no GxTx edges contained in it. The new sword design is shown in Figure 3.8c at right. The newly designed sword had a yield of 46.5%, so about a 10% improvement over the original.

The initial yield of the head invader array was 62.6%, which was much lower than has been reported in the literature (Petersen, Tikhomirov, and Qian, 2018). The structure of the array is shown in Figure 3.10a. We noticed that the head was forming six-tile structures, one of which is shown in Figure 3.10b. Because the head is a 2x2 array, it uses all four giving edges internally. Because it is an invader, it also has an external invading edge that matches one of its internal edges. One hypothesis for why the head was forming six-tile structures was that it was due to spurious interactions between the external invading edge and the matching internal receiving edge, though the correct tile should have an energetic advantage due to it correctly binding to two edges. Given the melting temperatures for one correctly bound edge T_M^1, as when the external edge binds to an internal edge, and two correctly bound edges T_M^2, as when the correct tile binds to the internal edge, for the array to form correctly during annealing $T_M^2 > T_M^1$. There is also the constraint that given the melting temperature of individual origami tiles T_M^0, $T_M^1 < T_M^0$ such that origami tiles form before the spurious interaction of an external edge with an internal edge does. We have previously observed that if $T_M^2 >> T_M^0$, then edge interactions may prevent tiles from forming correctly. However, in this case all tiles formed correctly, but were interacting spuriously. We therefore hypothesized that the condition that $T_M^1 < T_M^0$ was not being sufficiently met. Because T_M^1 and T_M^2 depend on interactions with the same receiving edge, the gap between them should be constant. Another way to think about it is that if the spurious interaction of the external edge with the internal edge were too strong, then the array could become kinetically trapped in the spuriously bound state. Thus, weakening the interaction could help prevent this.

We did not have control over the external edge because it needed all of its edges in order to invade at the tile displacement edge, but we dis have some control over the internal edges. We could remove staples and see if that helps collectively lower T_M^1 and T_M^2. We tried several edge code configurations to weaken the edge interactions: removing 3 outside edges, 2222222____, removing 3 inside edges, ____2222222, and removing 3 edges evenly dispersed, _2222_2222_, which has previously shown to improve the formation of 4-tile arrays Tikhomirov, Petersen, and Qian, 2017b. The invader designs are shown in Figure 3.10c, from left to right: the original design, inside edge design, the outside edge design, and the chosen coded edge design. The

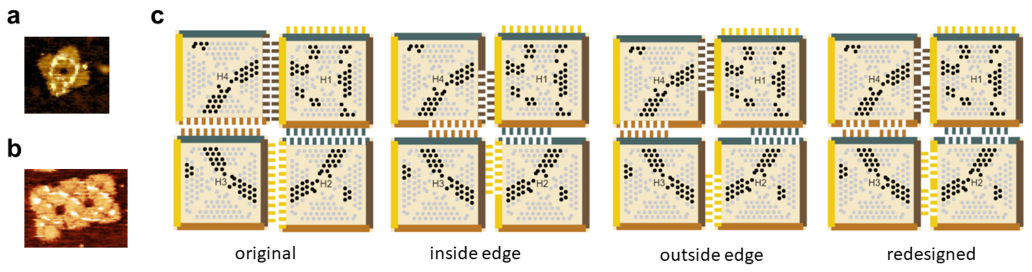


Figure 3.10: The head invader array. (a) An AFM image of the head array showing the details of its pattern. (b) A six-tile spurious head structure seen with the original design. (c) From left to right: the original head design, the inside edge design 2222222_____, the outside edge design _____2222222, and the coded edge design _2222_2222_. The coded edge design gave us the highest yield at 83.3%.

resultant yields were 17.6%, 43.8%, and 83.3%, respectively. We suspected that the low yield of the first two designs had to do with the structural instability of the arrays - in the first case, the tiles would be very flexible along the outer edges, resulting in large fluctuations and breathing in the edges. In the second case, the fluctuations would be less extreme as the hole is on the inside of the array, but if we think of edge interactions as remote toeholds, they were much more remote in that case. We chose the design in which we removed 3 edges evenly dispersed, with edge code _2222_2222_. Doing so weakened the interaction of the external edge with the internal, thus decreasing T_M^1. It was likely that $T_M^2 > T_M^0$ to begin with, so after the change it was likely that $T_M^2 \geq T_M^0$. This change resulted in an improvement of the yield to 83.3%, about a 20% increase over the original, as shown in Figure 3.11.

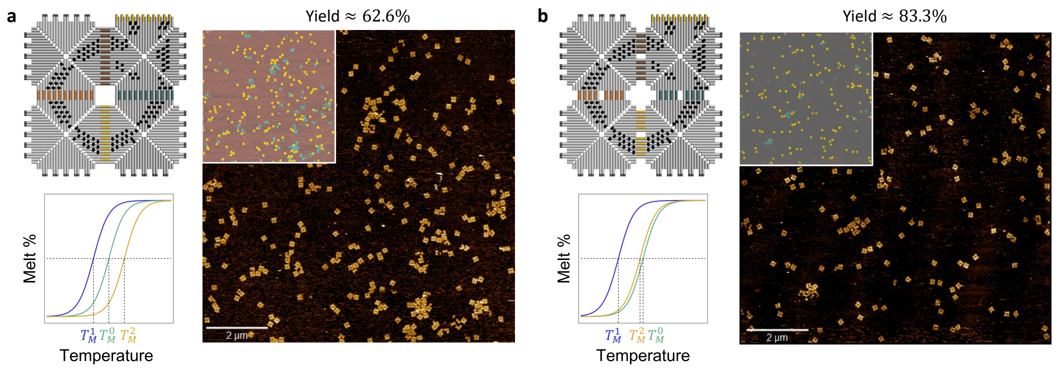


Figure 3.11: The head invader array yield. (a) The original edge and yield design. (b) The coded edge design, and improved yield. The adjusted melting curves provide insight into why the coded edges improve the design.

The tail invader was not a finite array, so its yield could not be quantified in the same way as the sword and head arrays. Rather, it formed a distribution of polymer lengths. Some example AFM images are shown in Figure 3.12, which shows that there are structures of varying lengths. We will discuss the analysis of this invader array in the next section, in which we explore some interesting emergent characteristics of the ability to grow in this manner.

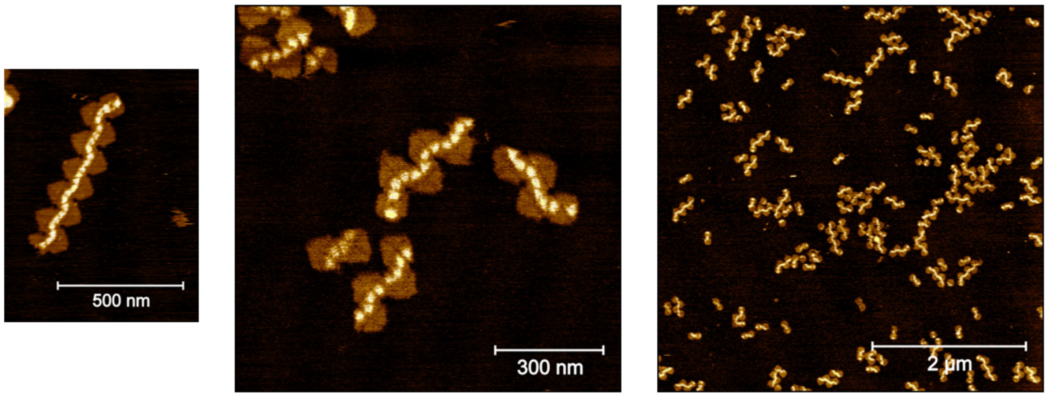


Figure 3.12: The tail invader array, shown at several scales. The tail forms varying sizes of array.

We did, however, realize that the tail invader contained G_xT_x edges, as shown in Figure 3.13a, specifically $G1T1$ and $G2T2$. Having determined that these kinds of edges may interfere with assembly formation as well as reaction rates, we decided to modify the tail design to avoid that type of edge. We thus redesigned the tail such that all edges were G_xT_y, but the property of polymer growth was preserved. The redesigned tail is shown in Figure 3.13b.

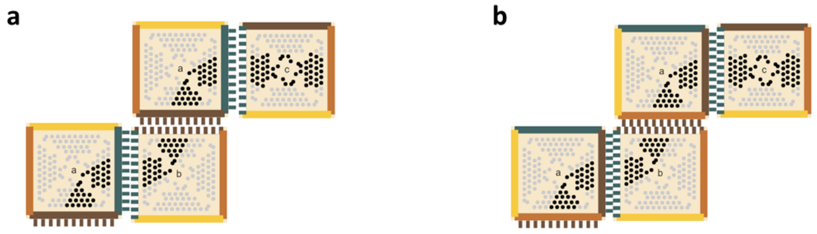


Figure 3.13: The redesigned tail invader array. (a) The original tail design, which contained G_xT_x edges. (b) The redesigned tail, in which all edges matched the desired G_xT_y pattern.

3.6 Tuning the length distribution of DNA origami polymers

The tail invader was designed to grow DNA origami polymers via the binding of two of its tiles, a and b. The third tile, c, was designed to cap the growth at one end. The other end of the polymer contains the active edge and was designed to remain free to perform tile displacement. The three tiles are shown in Figure 3.14a. When annealed together, the tiles formed tail structures that were a distribution of lengths. We investigated whether we could push the distribution towards growing longer polymers by tuning the ratios of the tiles used to anneal them. We hypothesized that if we annealed the tiles with a greater ratio of $(a + b) : c$, the relative decrease of capping tiles would result in longer polymer growth.

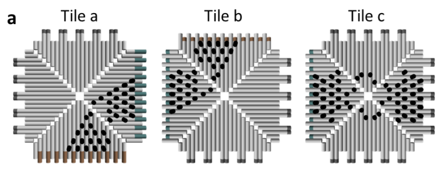
We chose the ratios of tiles to anneal first based on the ratio of $a : b$. The array was designed such that the active edge was on tile a, thus this edge needed to be exposed to perform the tile displacement reaction. Annealing the arrays with a ratio $a : b = (n + 1) : n$ would be most likely to result in an exposed a edge. We thus chose to test two different tile ratios $a : b : c$ in which we kept c constant at 1x, $3 : 2 : 1$ and $2 : 1 : 1$.

We analyzed AFM images to determine distributions of polymer lengths for each of the ratios. The patterns on the tiles, arcs on a and b and a straight line on c, guided us in classifying the polymers by their lengths, as shown in Figure 3.14b. We analyzed the distribution according to the formula:

$$f(s) = \frac{s * n(s)}{\sum_{s=1}^{n} s * n(s)}$$

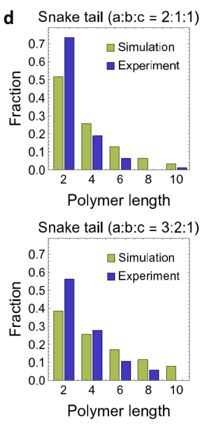
where s is the size of the structure in terms of number of tiles and $n(s)$ is the number of structures counted of that size.

We used a polymer CRN model to simulate the growth of the snake tail, as shown in Figure 3.14b. We began with three monomer reactants, a, b, and c. The rules for binding were (a, b), (a, c), and (b, a), given that a and b can bind at both edges. All binding reactions took place at the same rate k. Simulations showed us that using a lower ratio of tile c as compared to a and b would result in longer polymers; we confirmed that this was true experimentally. The simulations and experimental data are shown in Figure 3.14d.



b
$$
\begin{aligned}
&monomers = \{a,b,c\}\\
&rules = \{\{a,b\},\{a,c\},\{b,a\}\}\\
&rates = \{k,k,k\}\\
&maxlength = n\\
&reactions =\\
&\quad polymerCRN[monomers, rules, rates, maxlength]
\end{aligned}
$$

c
$$
\begin{array}{lll}
a + b \xrightarrow{k} ab & b + a \xrightarrow{k} ba & ba + b \xrightarrow{k} bab\\
a + c \xrightarrow{k} ac & b + ab \xrightarrow{k} bab & ba + c \xrightarrow{k} bac\\
ab + a \xrightarrow{k} aba & b + ac \xrightarrow{k} bac & a + ba \xrightarrow{k} aab
\end{array}
$$



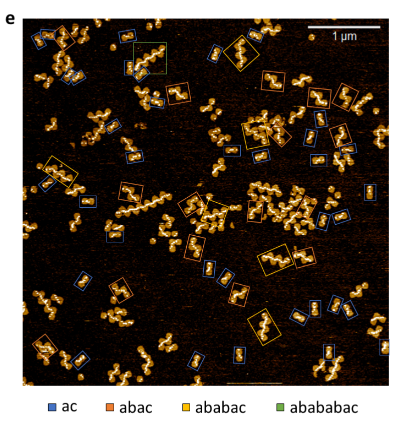


Figure 3.14: Tuning the length distribution of DNA origami polymers. (a) The tiles a, b, and c that form the snake tail polymer, with patterns shown. (b) The reactions with defined monomers, assembly rules, assembly rates, and maximum polymer length n. Though there is no maximum polymer length experimentally, in the simulation a maximum length was chosen that caused no noticeable changes in the distribution. (c) An example set of 9 reactions enumerated with $n = 3$. (d) Simulations and data showing that longer polymers form when there is less c as compared to a and b. A set of 570 reactions with $n = 20$ were used to simulate the system behavior. Simulations were performed with mass-action kinetics using the CRNSimulator (Soloveichik, 2009). Experimental data was obtained from two 5 by 5 μm AFM images. (e) An example AFM image with tile ratio $a : b : c = 3 : 2 : 1$.

Given the tile ratio $a : b : c = (n + 1) : n : 1$, we saw longer polymers grow when $n = 2$ than when $n = 1$. With the lower ratio of $c : (a + b)$, we saw fewer 2-tile structures and more structures that were 4-tile or longer. This confirmed that we could tune the ratios of $(a + b) : c$ to grow longer polymers.

The experimental data did show that there were many more 2-tile structures than the simulation predicted, as shown in Figure 3.14d. A two-tile structure is a structure ac that is unable to grow any larger due to the unavailability of additional tiles to react with. We hypothesized that this may be due to the formation of 3D structures by polymers without a tile c, e.g. ab, $abab$, etc. It has been shown previously that tiles that can bind to themselves will form 3D structures (Tikhomirov, Petersen, and Qian, 2018). This is possible because the flexibility along the seams of the tile allows bending to accommodate a 3D shape, and the unimolecular reaction of closing up into a 3D structure competes with the bimolecular reaction of adding another tile linearly. If 3D tube-like structures were forming, they would bend along the flexible seams of the tiles at 60° or 90° and form tubes with square or hexagonal cross-sections. as shown in Figure 3.15a, then they would be flattened by the AFM when imaged. This sequestering of tiles in 3D structures could explain the high concentration of polymer ac, as the active edges of those tiles would be unavailable to react with.

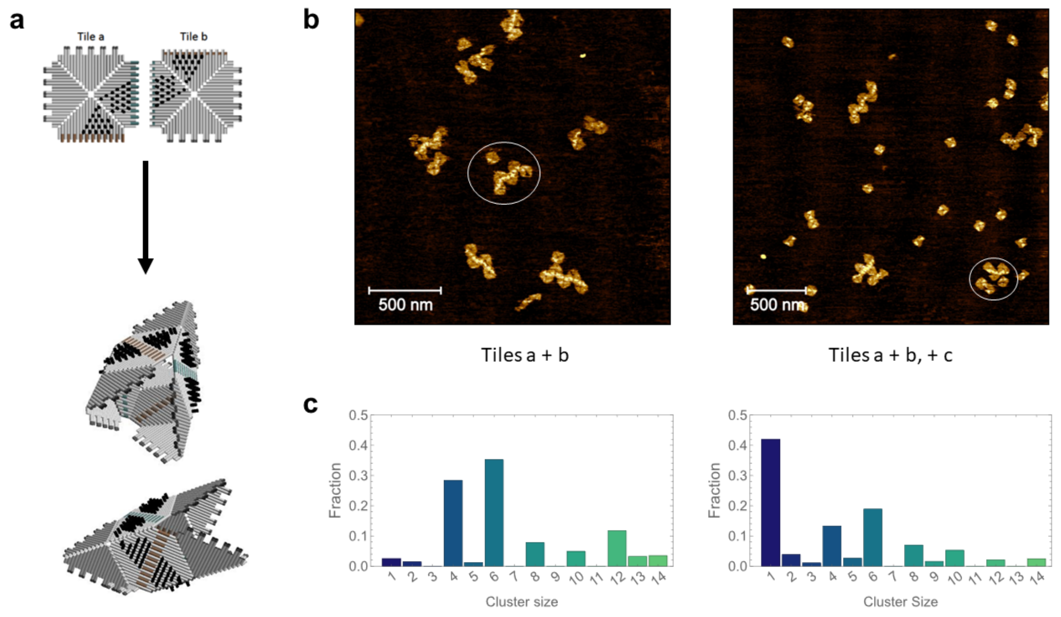


Figure 3.15: The tail invader forms 3D tubes. (a) Diagrams of the 3D structures that can form in the absence of tile c. See Figure 3.14a for tile diagrams. (b) Left, an example image of tiles $a + b$ annealed at a $1 : 1$ ratio. Right, an example image of tiles $a + b$ annealed at a $1 : 1$ ratio, with tile c added at $1x$ and allowed to hybridize for 24 hours. (c) Charts showing the percentages of structures of each polymer length for each sample.

We explored this hypothesis by annealing tiles a and b at a $1 : 1$ ratio in order to encourage 3D structures to form. We then added tile c to observe whether it would be able to bind. After 24 hours of hybridization, the tiles were imaged. As shown in Figure 3.15b, many tile c monomers in the $a : b + c$ case are free-floating, showing they were unable to bind to a polymer chain. There are about .38 more monomers than in the $a : b$ case, which is close to the 1/3 tile c that was added. We would expect the distribution of larger clusters in $a : b + c$ to be similar to $a : b$, just scaled by 2/3, which it is roughly. This supports the hypothesis that a and b were forming tubes with no available active edge. The few tile c that did bind could have diffused after the structures were broken open on the imaging surface.

3.7 3D to 2D reconfiguration

Having observed that 3D structures were forming in the snake tail, we next explored whether it would be possible to reconfigure the 3D structure into a 2D structure without having to flatten the structures on mica. Specifically, we wanted to use a tile displacement reaction to perform the reconfiguration. We modified one of the edges between tiles a and b to contain an open toehold that could be invaded by tile c. We created two versions of tile c: one that had its full invading edge, and a control version c_{ctrl} that had a truncated edge such that it would not be able to invade at the designed toehold. These tiles are shown in Figure 3.16a.

To confirm that 3D structures were still forming even with the addition of the toehold, we investigated the tube formation via fluorescence kinetics. We added a fluorophore to tile a and a quencher to tile b such that the fluorescence would be quenched if the tiles were bound. The positive control was a dimer annealed from versions of a and b, called a_{ctrl} and b_{ctrl}, that were inert on all edges except for the edges containing the fluorophore and quencher. These tiles are shown in Figure 3.16b, right. We used slightly different tile edges than in the tail invader to design tile $c4$ to be a giving edge invader as well as take advantage of fluorophore and quencher strands that we already had. The readout of this dimer should represent the fully bound system. The negative control was tile a, which contained the fluorophore, by itself. The experimental data, shown in Figure 3.16b, supported the hypothesis that all a and b were fully bound, suggesting that they were forming tubes even with the addition of the open toehold.

To investigate the reconfiguration, tiles a and b were annealed together at a $1:1$ ratio, then added to a well at $1x = 2nM$. Tile c or c_{ctrl} was added at $2x$ and the samples were allowed to hybridize for 24 hours. A diagram of the designed reaction is shown in Figure 3.16c, while the fluorophore and quencher placement are shown in d. Because 3D structures are opened up by the AFM tip during imaging, they can appear as clusters rather than bound arrays (Tikhomirov, Petersen, and Qian, 2018). The sample were imaged and analyzed by counting clusters based on their sizes. The samples without any tile c added mostly contained clusters containing an even number of tiles, as shown in a representative AFM image in Figure 3.16e, top left, and in the corresponding bar chart in Figure 3.16f. The samples with tile c revealed many shorter structures and monomers, as shown in the representative AFM images in Figure 3.16e, top right. There was a bias towards structures containing an odd number of tiles, particularly 3-tile structures, as shown in the corresponding bar

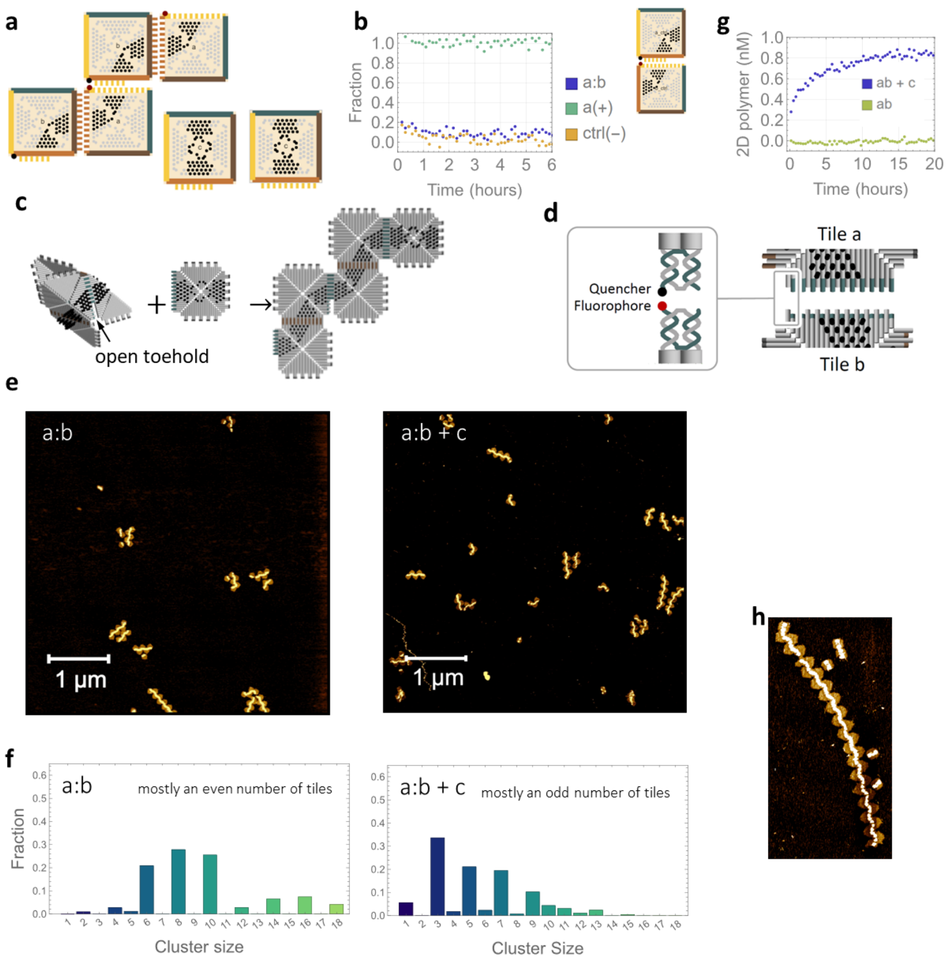


Figure 3.16: Reconfiguration from 3D to 2D. (a) Tiles a, b, c, and c_trl, where c and c_{ctrl} were added at 2x. (b) Left, fluorescence kinetics data showing that all tiles are fully bound when a and b are annealed $1:1$. Right, the ctrl(-) dimer. (c) A diagram of the 3D structure being tile displaced into a 2D structure. (d) The placement of the F/Q pair in this experiment. (e) Representative 5 micron AFM images of left, $a:b$, right, $a:b+c$. (f) The polymer length distributions, averaged from 4 5-micron images per sample. (g) Fluorescence kinetics data of the reaction. (h) The longest polymer we found over the course of doing these experiments, shown just for fun.

chart, suggesting that tile c was able to perform tile displacement on those samples. If tile displacement were to occur at every possible toehold, we would expect the sample to be all 3-tile structures. As there was an excess of tile c, it makes sense that there is also a fraction of monomers in this sample. Results for $a:b+c_{ctrl}$ are shown in Appendix A.

Just for fun, we have included one of the longest tail polymers that we observed in the course of performing these experiments in Figure 3.16h - 35 tiles long! This was an outlier in terms of length. Our hypothesis for the c_{ctrl} tile at the end of the polymer is that this was a 3D structure that broke open on the imaging surface after which a tile c_{ctrl} diffused to hybridize to it.

We experimented with adding in different concentrations of tile c to see if we could shift the distribution of cluster size. We found that as we reduced the ratio of tile c to $a : b$, we could produce larger 2D structures, as shown in Figure 3.17.

In addition to AFM experiments, we performed fluorescence kinetics experiments to confirm that the tile displacement reaction was happening as expected. We added a fluorophore to tile a and a quencher to tile b. If tile c were to successfully invade, we would observe an increase in fluorescence similar to the rates of tile displacement that we have previously observed. We did see this increase, suggesting that the 3D tubes were being tile displaced to become 2D structures. This data is shown in Figure 3.16g.

3.8 Sequential vs. simultaneous reconfiguration

We investigated the robustness of the tile displacement reaction by testing its speed and completion level given arrays with complex shapes. The full system with representative AFM images of each structure is shown in Figure 3.18. The starting sword array is a shape that had not been used before for these types of reactions. An array that forms a distribution of polymers had also not been used previously to perform tile displacement. We performed the reaction sequentially and simultaneously to compare its efficiency when one invader was present versus when both invaders were present, as well as evaluate any crosstalk between the invaders. To observe the reaction via fluorescence kinetics, we added a unique fluorophore to each of the two edges on the sword that are involved in the reaction such that we could observe both reactions simultaneously in orthogonal channels. We added quenchers on the tiles opposite the fluorophores. In this setup, once the invaders reacted with the sword, the quenchers would be displaced away, causing the fluorescence in both channels to increase. The positive control for this experiment was the annealed final array.

The data is shown in Figure 3.19a. The sword:head data, at left, has a k_{on} of $4.5 * 10^4 M^-1 s^-1$, while the sword:tail data, at right, has a k_{on} of $1.4 * 10^5 M^-1 s^-1$. These are both within the range of k_{on} that we observed in the dimer reactions in Table 3.1.

We also analyzed the yield by looking at AFM images of the samples after they had reached equilibrium. Here we analyzed yield by calculating

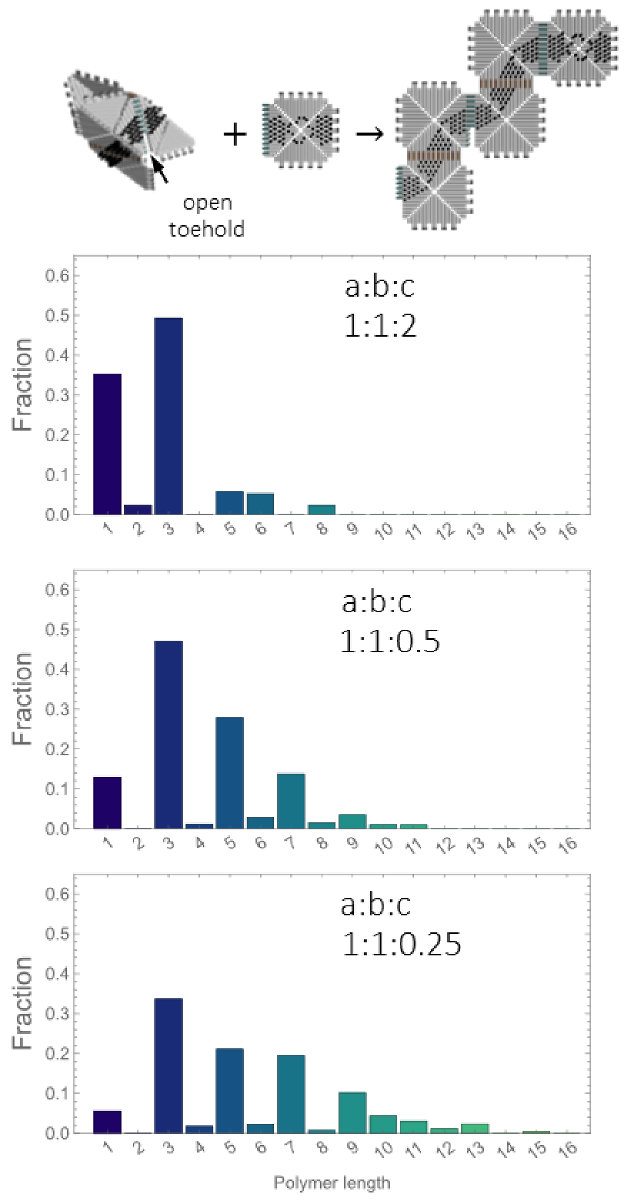


Figure 3.17: Producing larger tail structures. Top, a diagram of the reconfiguration from a 3D to a 2D structure. Below, polymer length distributions given varying tile c concentration as compared to $a : b$. As the ratio of tile c decreases, the resulting structures are larger.

$$\frac{n}{n+m}$$

where n is the number of products and m is the number of reactants in a given frame. We obtained an average yield of 40.3% yield of the sword:head reaction, averaged from 2 10um images and two different measures of the products, first

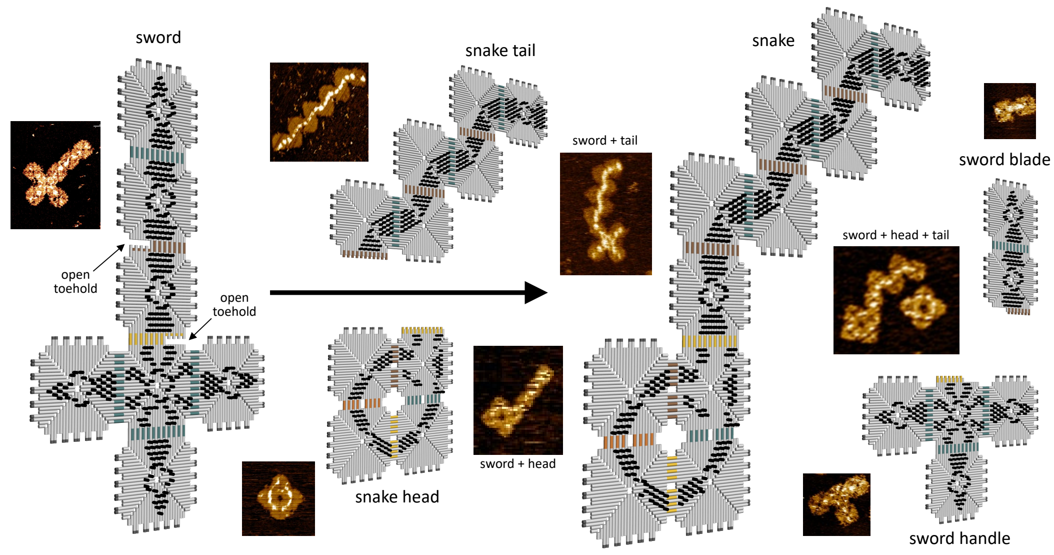


Figure 3.18: The full tile displacement system, shown with example AFM images of each type of array.

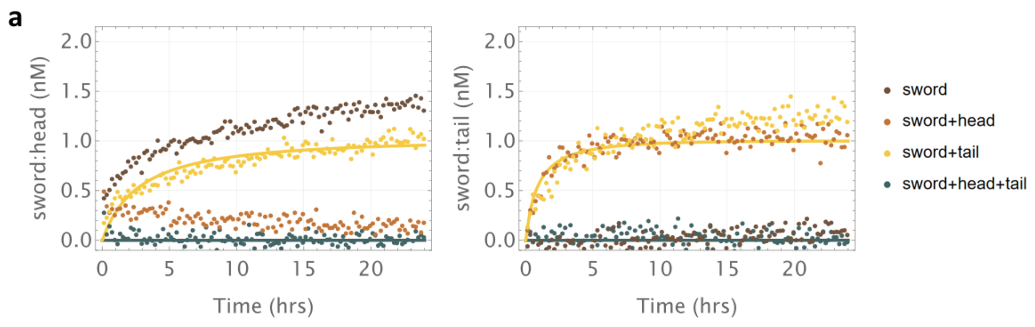


Figure 3.19: Sequential vs. simultanous reconfiguration. (a) Fluorescence kinetics data from an experiment in which the sword array was at 1x=2nM and invader arrays were added at 1.2x. Left, data from the fluorescent channel on the head side. The brown trajectory is the sword and head only, while the yellow trajectory is the sword, head, and tail mixed together. The reaction rates are roughly the same, but the completion level of the brown trajectory is higher, suggesting that in the presence of the tail invader, there is some spurious interaction that prevents the head from reacting as well with its tile displacement toehold. Right, data from the fluorescent channel on the tail side. The orange trajectory is the sword and the tail only, while the yellow trajectory is the sword, head, and tail mixed together. The similarity of the trajectories suggests that sequential and simultaneous reconfiguration on the tail side are comparable.

the sword:head product, then the hilt of the sword. For the sword:tail reaction, we measured an average yield of 96.8% from 2 10um images, using two separate measures for the products, first the sword:tail assembly, then the displaced blade of the sword. Finally, for the sword:head:tail reaction, we measured an average yield of 38.1% from 2 10um images. While the AFM data makes sense taken alone, in that the yield of the overall reaction is nearly equivalent to the yield of the lower yield sword:head reaction, it does not match the fluorescence kinetics data, which suggests that both reactions had an equal yield. Analyzing data from AFM images can be notoriously unreliable, particularly if care is not taken to analyze enough data to account for variation between images. However, there is rarely this stark of a discrepancy between the AFM data and the fluorescence kinetics data.

One explanation could be that the products of the sword:head reaction are forming aggregates that we did not happen to find when imaging the sample. However, it seems unlikely that both the hilt of the sword and the sword:head assembly are aggregating, particularly considering that we saw hilts and sword:head complexes in the images we took. Another possibility is that there is another reaction pathway that results in those products reconfiguring once again after the initial reaction takes place. One form of this could be that the head invader dissociates from the rest of the sword after displacing the hilt, which would result in us seeing the reaction take place in the fluorescence kinetics data but not as many sword:head assemblies in AFM images. In some images we saw significantly more hilt assemblies than sword:head assemblies, so this could be a plausible explanation.

We imaged the positive control sample as a comparison, in which all tiles in the expected final structure were directly annealed together. We saw that there were unbound head invaders, suggesting that there may be an experimental issue with the staples at the active edge of the head invader, or the receiving edge on the sword.

3.9 Discussion

We have shown that tile displacement is a robust type of reaction that can be used to reconfigure assemblies of complex shape, not just square arrays. We have also shown that the reconfiguration can be done via invader arrays that have complex shapes. We can improve array formation by redesigning edges to match $GxTy$, as well as by introducing coded edges when there is competition between an outside invader edge and an internal edge. We have expanded the design space for tile displacement reactions without sacrificing the reaction speed by determining that toehold and branch migration domains can be different configurations – e.g., the toehold can be a giving edge and the branch migration domain can be a receiving edge – and that the toehold can be discontinuous. We have also shown that invaders that are G_xT_x, or their sticky edge extensions match the triangle edge they are being extended from, are slower than invaders that are G_xT_y, likely because the transient branch migration at each of the 22 sticky ends and stacking bonds cause the invaders to sequester each other.

In tile assemblies in which two of the tiles can bind to each other at two distinct edges, we have shown that we can grow a distribution of polymer lengths. By adjusting the ratios of the tiles, we can promote the growth of longer polymers. We have also shown that such assemblies form 3D structures in the absence of a capping tile. We demonstrated that it is possible to use tile displacement to reconfigure those 3D structures into 2D structures.

We showed that we could obtain similar, if not identical, reaction rates when reconfiguring a complex-shaped assembly with one complex invader at a time, or with both together. This suggests that the tile displacement reaction is robust to the size and shape of the array to be reconfigured and the invader arrays. This conclusion opens up the possibilities of the kinds of reconfiguration events that could be driven by tile displacement reactions. A unique aspect of this kind of reaction is the capability we have to build other types of systems on top of origami tiles, such as the robot discussed in Chapter 2. Theoretically, tile displacement is Turing universal and can simulate arbitrary two-dimensional synchronous block cellular automata (Winfree and Qian, 2022). If these behaviors can be implemented experimentally, then it may be possible to use complex computation to modulate a changing environment for a device, circuit, or robot that is built on top of multi-origami structures.

Chapter 4

A LOSER-TAKE-ALL DNA CIRCUIT

4.1 Introduction

DNA circuits have been developed to perform complex molecular information processing tasks, including thresholding, signal amplification, Boolean logic, and neural network computation (David Yu Zhang et al., 2007; Seelig et al., 2006; Qian and Winfree, 2011; Cherry and Qian, 2018). They are well-suited for such tasks because of their programmability and ability to interface with a wide variety of inputs and outputs, including DNA, RNA, and proteins. They are also simple and scalable because they depend mainly on two- or three-stranded molecules that perform well-characterized reactions.

The underlying mechanism of toehold-mediated strand displacement that is used as the basis for these technologies was first demonstrated to power molecular tweezers that could transition between different states (Yurke, Turberfield, et al., 2000). Toehold-mediated strand displacement has since been thoroughly studied and characterized. It is now a well-understood mechanism that can be controlled over six orders of magnitude via the engineering of the toehold domain (David Y. Zhang and Winfree, 2009; Srinivas et al., 2013).

It has been shown both in theory (Lakin and Stefanovic, 2016) and in practice (Qian, Winfree, and Bruck, 2011; Cherry and Qian, 2018). that it is possible to build DNA circuits that perform neural network computation. A Hopfield associative memory was implemented using DNA strand displacement cascades that could recognize an incomplete pattern and recall the complete one that it best matched. The complexity of the input patterns in this case were limited by the need to use a dual-rail technique to represent negative gates as well as noise in the feedback loops that required many signal restoration gates to suppress (Qian, Winfree, and Bruck, 2011). The implementation was limited to a 4-bit pattern. Later, a winner-take-all neural network was built using DNA strand displacement cascades that did not need to use dual-rail to represent negative weights and did not need feedback loops. The simpler implementation of this network versus the Hopfield associative memory enabled the molecular classification in the winner-take-all network to be scaled up to a 100-bit pattern.

One framework that has proven particularly adept at supporting complex DNA circuit function, including that of the winner-take-all neural network previously mentioned, is seesaw circuits. They are a simple architecture that is easily composable to build cascades. The seesaw gate motif is illustrated in Figure 4.1.

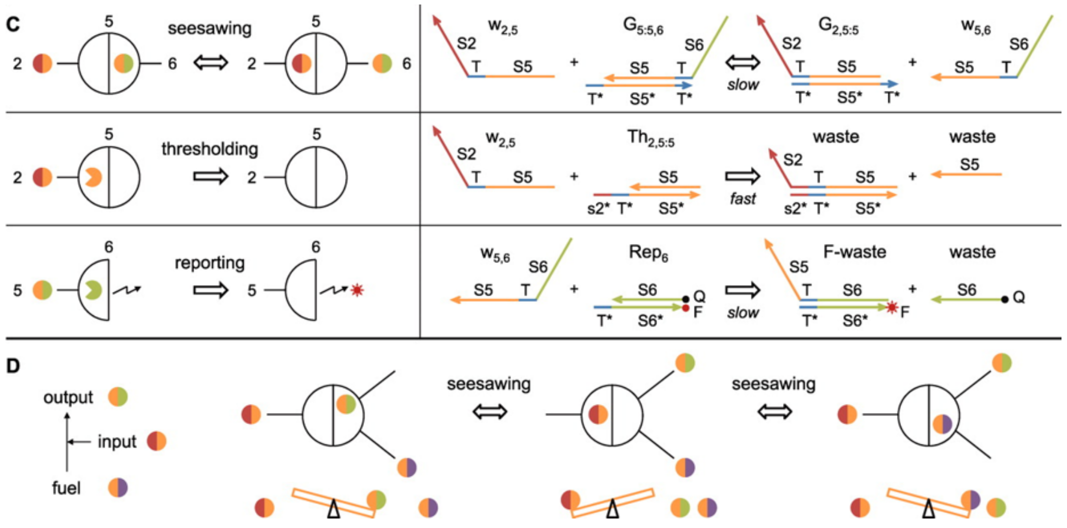


Figure 4.1: Seesaw gate motif. (C) Three basic reaction mechanisms involved in a seesaw network: seesawing, thresholding, and reporting. Solid circles with two colors indicate signal strands that have two sides. Colored pac-men indicate threshold or reporter complexes. w2,5 is the signal strand that connects gates 2 and 5; G5:5,6 is signal strand w5,6 bound to gate 5; Th2,5:5 is the threshold that absorbs w2,5 when it arrives at gate 5; and Rep6 is the reporter that absorbs wi,6 and generates fluorescence signal for any i. (D) One cycle of a seesaw catalytic reaction. Adapted from Qian and Winfree, 2011.

4.2 System Design

We took inspiration from the winner-take-all neural network in designing a loser-take-all DNA circuit that also uses the seesaw gate motif in its implementation. It computes which input is the smallest among 3 inputs. When connected to a winner-take-all network, the resulting neural network would be able to compute which pattern is least like a memory, as shown in Figure 4.2. This computation may be useful in some circumstances in which uniqueness of a pattern is important, such as in outlier removal, or when it is difficult for the previously demonstrated winner-take-all network to distinguish between multiple patterns that are similar to the memory. The loser-take-all circuit expands the functional capabilities of DNA-based neural networks by introducing an architecture that is composable with the winner-take-all implementation.

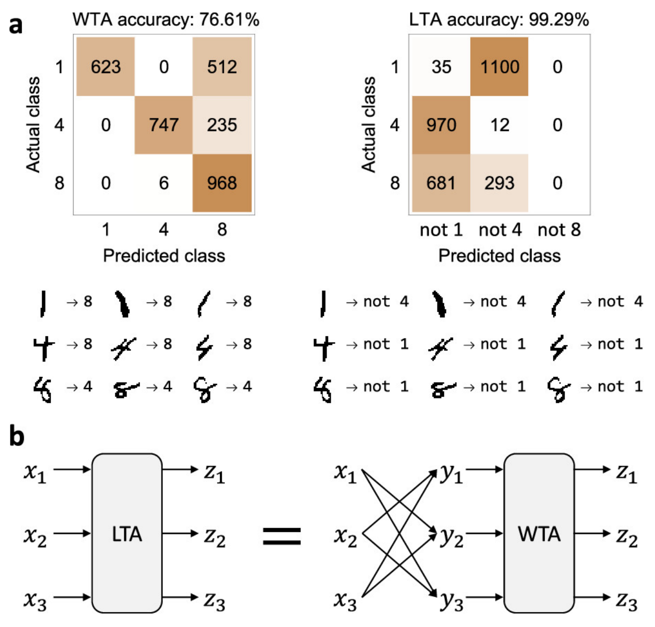


Figure 4.2: Concept of a loser-take-all circuit. (a) Confusion matrix and example pattern classification results of winner-take-all (WTA) and loser-take-all (LTA) neural networks. Training and testing patterns were taken from the MNIST database (12) and converted from grayscale to binary. Weights were assigned as the average of the first hundred patterns in the training data set. (b) Abstract design of a three-input loser-take-all circuit. From (Rodriguez, Sarraf, and Qian, 2021).

The usefulness of the loser-take-all circuit hinges on the fact that negative weights are not allowed in the winner-take-all implementation. Thus, there is no way that signals can be reversed in order to compute the loser-take-all function. It is also not possible to use an annihilator to reverse the signal, as this would require that the 1 in the $1 - x_i$ annihilation reaction would need to be present before x_i were to arrive. The 1 would be consumed by the downstream winner-take-all layer before x_i arrived, which would result in an incorrect output.

In the loser-take-all circuit, signal reversal is performed via the following operation, which for each input signal computes the average all input signals excluding itself:

$$y_i = \sum_{j \neq i} \frac{x_j}{n-1}$$

where $x_i \in \Re, i, j \in 1, 2, \ldots, n$ for a circuit with n inputs. This will result in the largest y_i corresponding to the smallest x_i. Then, the winner-take-all function is computed on the reversed input signals:

$$z_i = \begin{cases} 1, & \text{if } y_i > y_j, \forall j \neq i \\ 0, & \text{otherwise} \end{cases}$$

where $x_i \in \Re, i, j \in 1, 2, ..., n$ for a circuit with n inputs.

4.3 Methods

Sample Preparation

DNA oligonucleotides were purchased from Integrated DNA Technologies (IDT). Reporter strands with fluorophores and quenchers were ordered with high-performance liquid chromatography (HPLC) purification, while gate, fuel, annihilator, and input strands were ordered unpurified (standard desalting). Strands were shipped with formulation service LabReady (100 μM in IDTE buffer at pH 8.0). They were stored at 4.0 °C.

Annihilator and gate complexes were annealed at 45 μM with a $1:1$ ratio of top and bottom strands. Reporters were annealed at 20 μM with a $1.2:1$ ratio of top and bottom strands. All complexes were annealed in TE buffer with 12.5 mM Mg^{2+}. Annealing took place in a thermocycler (Eppendorf). Samples were heated to 90 °C for 5 min and then cooled to 20 °C at a rate of 0.1 °C per 6 s.

As the excess top strands of the reporter complexes do not interfere with designed molecular interactions in the circuit, reporter complexes do not need to be purified. However, an excess of either the top or bottom strands of the annihilator or gate complexes would affect the circuit behavior. Therefore, annihilator and gate complexes were purified using 12% polyacrylamide gel electrophoresis (PAGE). The gels were run at 150 V for roughly 6 h. Bands containing the complexes were cut out from the gel, diced into smaller pieces, and incubated for at least 24 h at room temperature in TE buffer with 12.5 mM Mg^{2+}. The buffer containing each complex that diffused out from the gel pieces was then collected. The absorbance of each collected sample at 260 nm was measured using a NanoDrop (Thermo Fisher). Along with the extinction coefficients of the complexes, these data were used to calculate the concentration of each complex.

Fluorescence Kinetics Experiments

Fluorescent data was collected on a microplate reader (Synergy H1, Biotek). A 96-well plate (Corning) was used for experiments, with 110 μL of reaction mixture per well. The standard concentration for the experiments was 50 nM. Excitation/emission wavelengths were 496 nm/525 nm for fluorophore ATTO488, 555 nm/582 nm for fluorophore ATTO550, and 598 nm/629 nm for fluorophore ATTO590. Readings were taken every 2 min for the duration of the experiment.

4.4 Experimental Implementation

The DNA implementation of the circuit is shown in Figure 4.3.

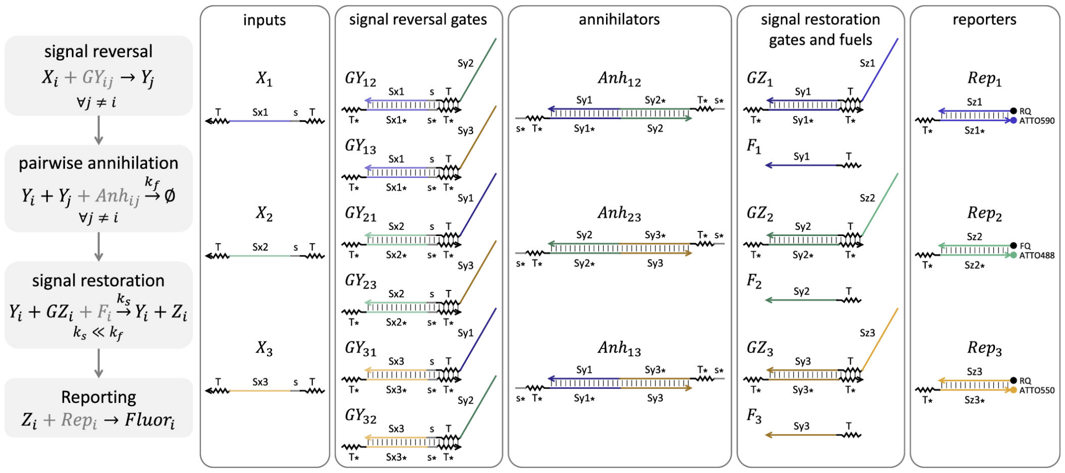


Figure 4.3: DNA strand-displacement implementation of a three-input loser-take-all circuit. In the chemical reactions, the species in black or gray are needed as part of the function or to facilitate the reactions, respectively. The concentrations of facilitating species are in excess. The concentration of a signal strand corresponds to the value of a variable (e.g., x1 = [X1]). Signal Y_j is the union of all top strands in GY_{ij}. Signal Z_i is the top strand in GZ_i. The initial concentration of GZ_i (e.g., $[GZ_i]_0$ = 1 standard concentration) determines the steady-state concentration of $Fluor_i$ when output Z_i is computed to be ON. Zigzagged lines indicate toehold domains and straight lines indicate branch migration domains. Extended toehold domains on annihilators are indicated as s* T*. Clamp domains for reducing leak between double-stranded complexes are not shown here but included in Figure S1. Three distinct ATTO dyes were used in reporters for fluorescence readout. From (Rodriguez, Sarraf, and Qian, 2021).

To confirm that we could implement signal reversal, we began by connecting the signal reversal gates directly to the reporters. We wanted to check two things: (1) that all input strands reacted with the signal reversal gates at the same rate, and (2) that the inputs would produce evenly split outputs. To better ensure that (1) was met, the signal reversal gates were designed to have the same toehold sequence. The largest difference between the pairs of output signals tested with each input to test (2) was 10%. As shown in Figure 4.4, the concentrations of the outputs produced by input X_1 were about 40% lower than the predicted value of $1x$, which could indicate that the effective concentration of that input was lower than expected. Figure 4.4 shows a successful demonstration of signal reversal when all three inputs and six gates were mixed together, with two different input signal combinations.

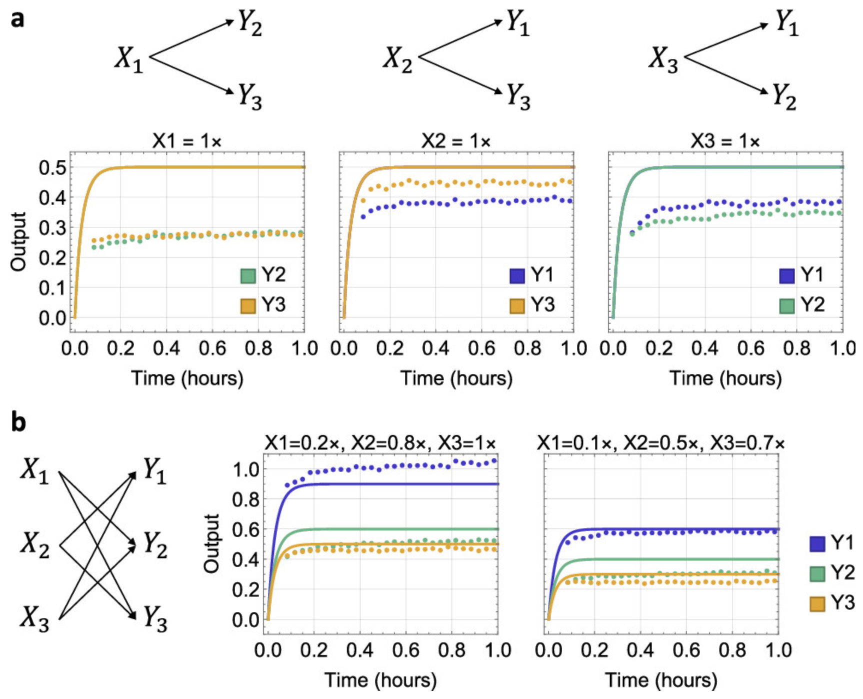


Figure 4.4: Demonstration of signal reversal. Simulations are shown as solid lines, and experimental data are shown as dotted lines. (a) Input strands reacting with a pair of signal reversal gates, with standard concentration $1x = 50nM$. The effective concentration of X_1 is likely lower than expected, resulting in a lower production of Y_2 and Y_3 than expected. The same could be true for X_3, though to a lesser extent. (b) Demonstration of signal reversal given two different combinations of the three inputs. In both cases, the lowest input concentration corresponds to the highest output concentration. Gate and reporter concentrations were $2x$. Adapted from (Rodriguez, Sarraf, and Qian, 2021).

An important factor for accurate signal reversal is that all gates have equal reaction rates. To promote this, we used common toeholds on the signal reversal gates as well as on the annihilators. The remaining factor that could influence reaction rates was the branch migration domains. To investigate the differences in rate that these caused, we measured the rates of signal restoration gates, which could be directly connected to reporters and have the same branch migration domains as annihilators. The slowest strand displacement rate ks_i was $5 * 10^4 M{-}1s{-}1$, while the largest was $12 * 10^4 M{-}1s{-}1$, indicating a 2.4-fold difference in rates, as shown in Figure 4.5a. Adding in the signal reversal gates and annihilators allowed us to calculate the toehold dissociation rate kr_i, as shown in Figure 4.5b.

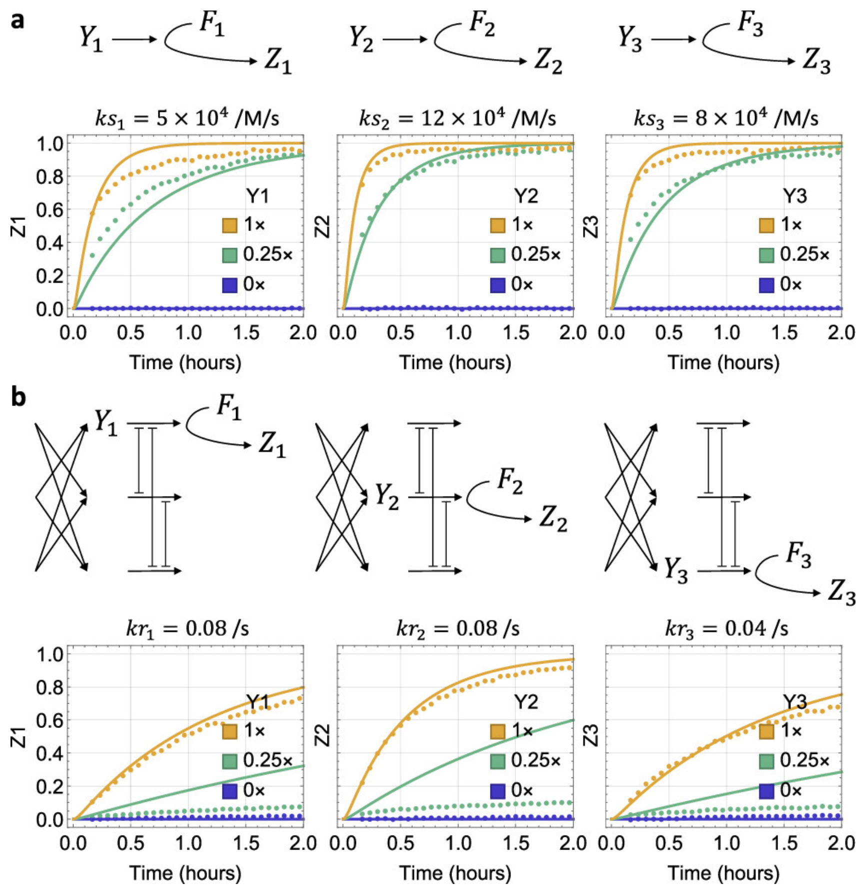


Figure 4.5: Rate measurements in signal restoration. (a) Signal restoration gates connected directly to reporters, which allowed us to calculate the strand displacement rate ks_i. (b) Signal restoration with signal reversal gates and annihilators, which allowed us to calculate kr_i. $1x = 50nM$ and concentrations of signal reversal gates, annihilators, signal resotration gates, fuels, and reporters were $2x$, $4x$, $1x$, $2x$, and $2x$, respectively. Adapted from (Rodriguez, Sarraf, and Qian, 2021).

4.5 Debugging the circuit

The rate estimations for ks_i and kr_i in the previous section, when input into simulations, predicated that the circuit would be biased towards turning Z_2 on the fastest. We confirmed this experimentally by showing that Z_2 turned on faster than Z_1 or Z_3 in analogous input conditions, as shown in Figure 4.6a. Rather than undertake the challenging task of redesigning branch migration sequences to be more balanced, we instead tried adjusting concentrations to compensate for the observed rate differences. Simulations confirmed that if we were to reduce the concentration of annihilator Anh_{13} by half, we would introduce a bias against producing Z_2 that would equalize the preexisting bias towards producing Z_2.

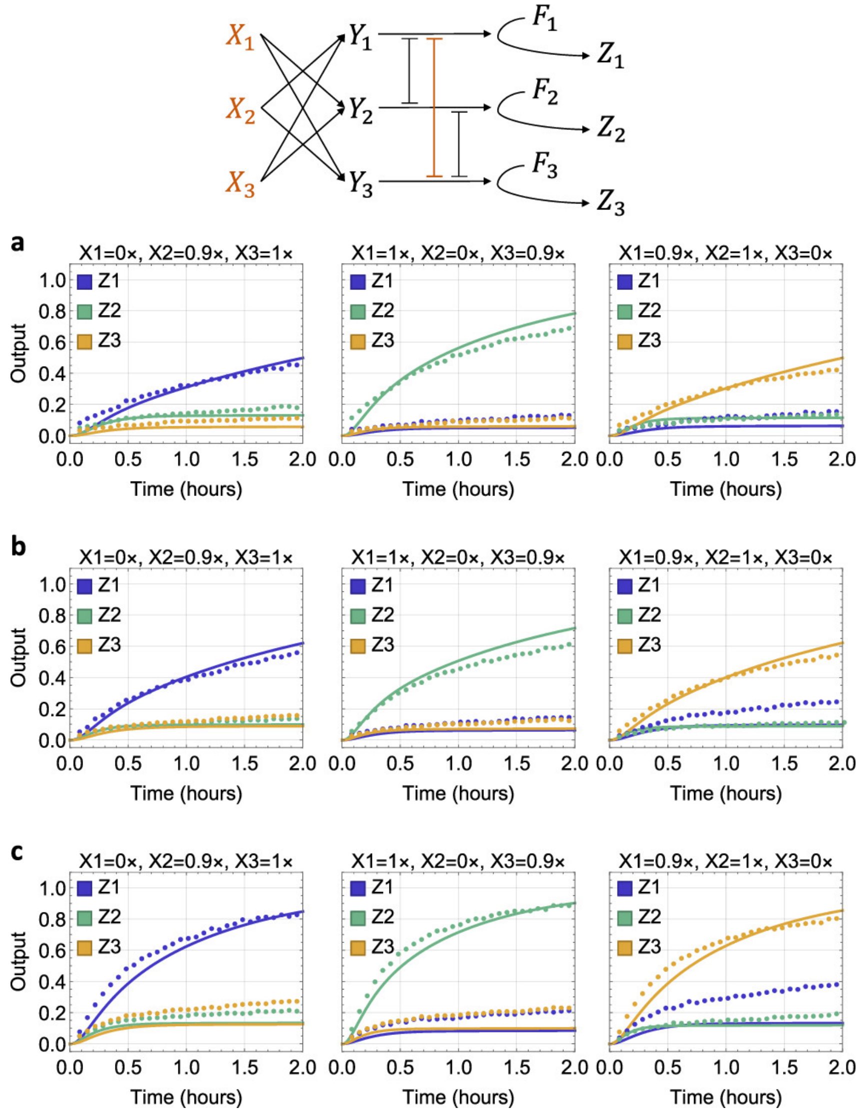


Figure 4.6: Adjusting annihilator and input concentrations. (a) The three-input loser-take-all circuit. (b) The circuit after adjusting the annihilator concentration of Anh_{13}, which shows more balanced reaction rates. (c) The circuit after adjusting the input concentrations, which results in a speedup in all reaction rates. $1x$=50nM and concentrations of signal reversal gates, annihilators, signal resotration gates, fuels, and reporters were $2x$, $4x$, $1x$, $2x$, and $2x$, respectively. In (b) and (c), $Anh_{13} = 2x$. In (c), input concentrations were increased to 100 nM. Adapted from (Rodriguez, Sarraf, and Qian, 2021).

We ran this experiment and the result supported the hypothesis. When the annihilator concentration was adjusted to 2x from 4x, the kinetics of all three outputs were similar in the same input conditions that originally revealed the bias, as shown in Figure 4.6b. We can generalize this strategy to say that when the production of any output Z_i is faster than the production of the other outputs, reducing the annihilator concentration for other outputs besides Z_i will help balance the rates. This is true because the this reduces the competition between the other input signals, thus increasing their effective reaction rates. We kept this concentration adjustment for all future experiments.

In the previous section, we considered that the input strands were likely at a lower effective concentration than expected, particularly X_1 and slightly less so X_3, due to their generating less output that expected. Simulations confirmed that doubling the input concentration would speed up the circuit and result in better separation between ON and OFF states. To that end, we increased the input concentration from 1x=50nM to 1x=100nM. The experimental results supported this hypothesis as well, as shown in Figure 4.6c. We kept this adjustment for all future experiments.

4.6 A three-input LTA circuit with nine unique input combinations

We demonstrated that the circuit could produce the correct output given nine different input combinations, as shown in Figure 4.7. In some cases the separation between ON and OFF was not as large as expected, though in many cases the ON trajectory was faster than the simulation predicted. The circuit was still able to correctly compute the LTA function in all nine cases, though the system behavior warranted further investigation.

We considered another possible explanation for the circuit behavior, which was based on the design of the input strands. As compared to the winner-take-all DNA neural network, in which the input strands all had one toehold, in this design the input strands have two toeholds. We hypothesized that having two toeholds on the inputs, which make the reaction with the signal reversal gates irreversible, might result in augmented spurious interactions with the annihilators given the universality of the toehold. To investigate this possibility, we removed one toehold from the input strands, as shown in Figure 4.8a. This resulted in reduced separation between the ON and OFF states, as shown in Figure 4.8b. This suggested that the lack of irreversibility introduced crosstalk among the signal reversal gates, perhaps by allowing them to leak with the annihilators via the universal *s* domain.

A second strategy was to change one of the toeholds on the input strands to be unique from the universal toehold, labeled T2 in 4.9a. We hypothesized that this would prevent crosstalk between the input signals and the annihilators via that toehold.

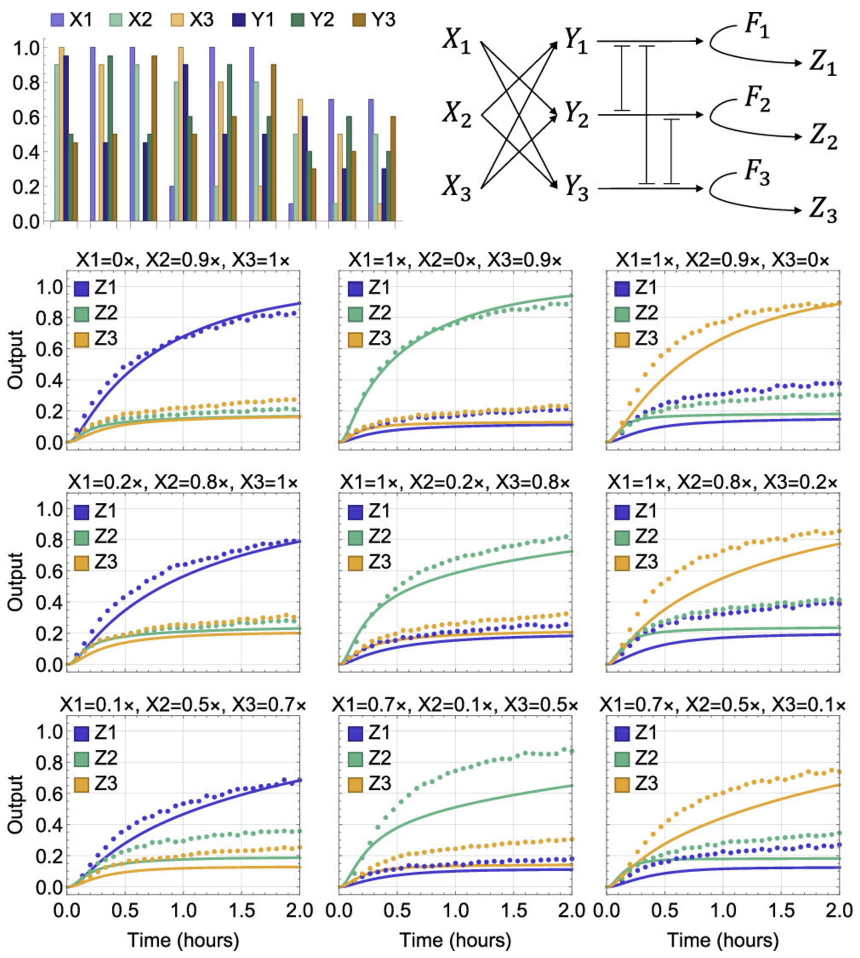


Figure 4.7: Three-input LTA circuit with nine unique input combinations. The circuit correctly computes the LTA function, though the separation between ON and OFF is not as large as expected in some cases. Simulations are shown as solid lines, while experimental data are shown as dotted lines. $1x$=100nM for input concentrations, while $1x$=50nM for all other molecules. Annihilators were at $4x$ except for Anh_{13} at $2x$. Signal reversal gates, signal restoration gates, fuels, and reporters were at $2x$, $1x$, $2x$, and $2x$, respectively. Adapted from (Rodriguez, Sarraf, and Qian, 2021).

The circuit behavior improved slightly, as shown in Figure 4.9b, though it was not as significant of an imporvement as was expected.

4.7 Discussion

Here we demonstrated that we could build a simple and scalable 3-input loser-take-all circuit that computes the output corresponding to the lowest input signal. There were some discrepancies in strand displacement rates due to differences in branch migration rates, which were partially corrected by adjusting annihilator and input

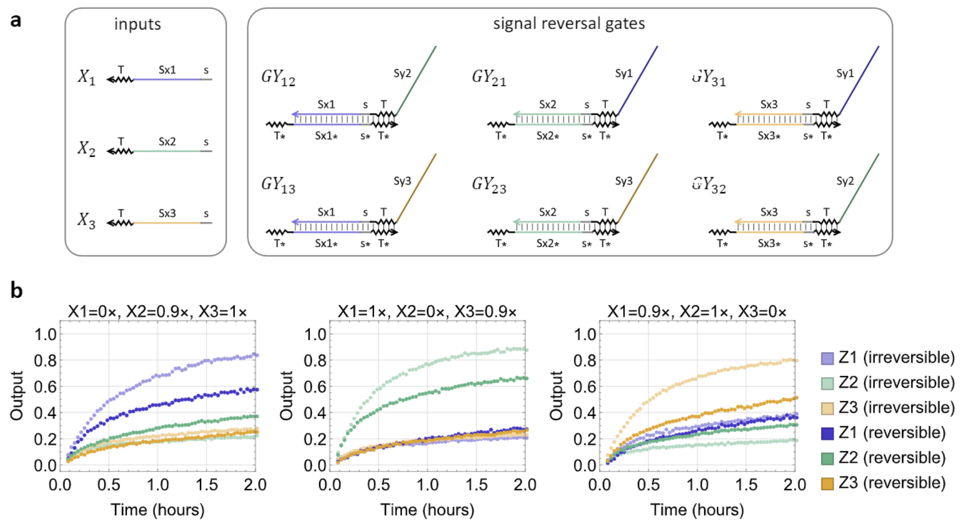


Figure 4.8: Reversible signal reversal. (a) Diagrams showing the inputs with their second toehold T removed. (b) Fluorescence kinetics data showing that system behavior worsened as compared to the data from when the signal reversal was irreversible, shown in lighter colors on the same graph. $1x$=100nM for input concentrations, while $1x$=50nM for all other molecules. Annihilators were at $4x$ except for Anh_{13} at $2x$. Signal reversal gates, signal restoration gates, fuels, and reporters were at $2x$, $1x$, $2x$, and $2x$, respectively. Adapted from (Rodriguez, Sarraf, and Qian, 2021).

concentrations, though they could not be completely balanced via these strategies. We removed the second toehold on the input strands, which worsened the system behavior, then tried changing the first toehold to be different than the annihilator toeholds, which improved the circuit marginally.

It may not be possible to fully correct the system behavior without addressing differences in strand displacement rates due to branch migration sequences. Though we have some understanding of how to design sequences to have similar branch migration rates – e.g., by matching the number and dispersion of CG pairs – it is still difficult to design sequences that are orthogonal and have similar branch migration rates. There is research in progress to make this process of finding orthogonal sequences with balanced rates more efficient. Hopefully as those tools become available it will be easier to design a more robust loser-take-all circuit, or any circuit that is heavily dependent on balanced reaction rates.

The answer to this dilemma could be to design circuits that are robust to 2-fold variations in reaction rates and up to 40% differences in signal concentrations. There have

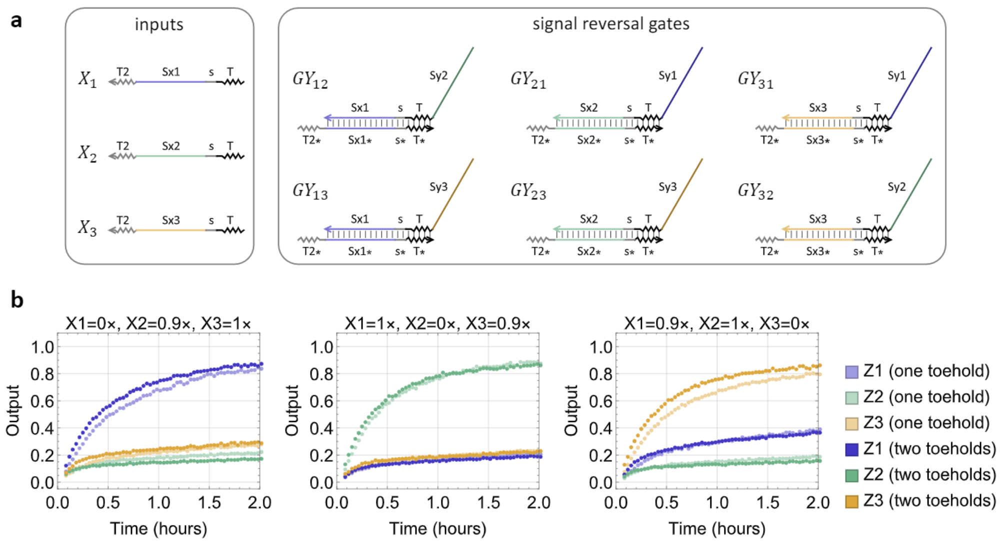


Figure 4.9: Unique toeholds in signal reversal. (a) Diagrams showing T_2 on the input strands. (b) Fluorescence kinetics data showing a slightly improvement over system behavior in which both toeholds were T. The previous data is shown in lighter colors on the same graph. $1x$=100nM for input concentrations, while $1x$=50nM for all other molecules. Annihilators were at $4x$ except for Anh_{13} at $2x$. Signal reversal gates, signal restoration gates, fuels, and reporters were at $2x$, $1x$, $2x$, and $2x$, respectively. Adapted from (Rodriguez, Sarraf, and Qian, 2021).

been demonstrations of DNA circuits that are robust to such variations (Thubagere et al., 2017; Qian and Winfree, 2011). Another example is thermodynamic binding networks, which have been explored theoretically, in which reaction pathways are consistent with thermodynamic equilibrium, rather than relying on kinetics (Doty et al., 2017).

The loser-take-all circuit introduces the capability of recognize noisy patterns if composed together with a winner-take-all neural network, particularly those that may be indistinguishable by the winner-take-all function. This could be a particularly useful tool for noisy in vivo environments. This function would be an interesting modulator for molecular robot behavior.

Chapter 5

DNA STRAND-DISPLACEMENT TEMPORAL LOGIC CIRCUITS

5.1 Introduction

The importance of relative timing in biological systems has been established in processes as diverse as the regulation of gene expression (Lin et al., 2015) to echolocation and how the brain interprets odor stimulation (Edwards, Alder, and Rose, 2002; Haddad et al., 2013). Being able to determine the relative timing of environmental signals could be a useful type of control for DNA-based systems.

There have been theoretical explorations and experimental implementations of molecular circuits that can determine the order in which inputs arrived. In theory, it has been shown that DNA strand displacement reactions can be used to implement temporal logic (Lakin and Stefanovic, 2017), recognize temporal patterns (O'Brien and Murugan, 2019), and record the order in which n events occur (Cardelli, 2021). Experimentally, circuits that process temporal information have been implemented in synthetic biology (Friedland et al., 2009; Hsiao et al., 2016). There have also been DNA circuits that can record timing information (Kishi et al., 2018) and use a strategy called cross-inhibition in which the first input to arrive inhibits the pathways of all subsequent inputs (C. Liu et al., 2020). This strategy is effective in determining the first input to arrive, much like a winner-take-all network determines the best-matched input. However, it does not provide additional information about subsequent input signals and would produce the same output whether or not other inputs ever arrived at all. Thus, it can provide n distinct outputs for n inputs.

In contrast, temporal memory is a strategy in which the circuit remembers previous input information and only produces an output when all input signals have arrived. Thus, there are $n!$ possible outputs when there are n inputs. We chose to implement this strategy using a DNA strand-displacement circuit.

A thorough understanding of the three-way strand displacement reaction that powers such motifs has been developed, which has enabled efficient engineering of DNA strand-displacement circuits (Yurke and Jr., 2003; David Y. Zhang and Winfree, 2009; Srinivas et al., 2013). Previously designed DNA strand-displacement motifs, such as seesaw gates (Qian and Winfree, 2011) and cooperative hybridization (David Yu Zhang, 2010), have enabled the construction of circuits that perform complex information-processing tasks, including thresholding, signal amplification, Boolean logic, and neural network computation (David Yu Zhang et al., 2007; Seelig et al., 2006; Qian and Winfree, 2011; Cherry and Qian, 2018). Previously developed DNA strand-displacement motifs, such as seesaw gates and cooperative hybridization, have enabled the construction of circuits that perform complex information-processing tasks. Such tasks have included thresholding, signal amplification, Boolean logic, and neural network computation (David Yu Zhang et al., 2007; Seelig et al., 2006; Qian and Winfree, 2011; Cherry and Qian, 2018).

To build this temporal memory circuit, we used both seesaw gates and cooperative hybridization. We show the basic mechanism for seesaw gates in Chapter 4. Cooperative hybridization is a mechanism in which two strands must simultaneously bind to a molecule in order for the reaction to proceed, as illustrated in Figure 5.1.

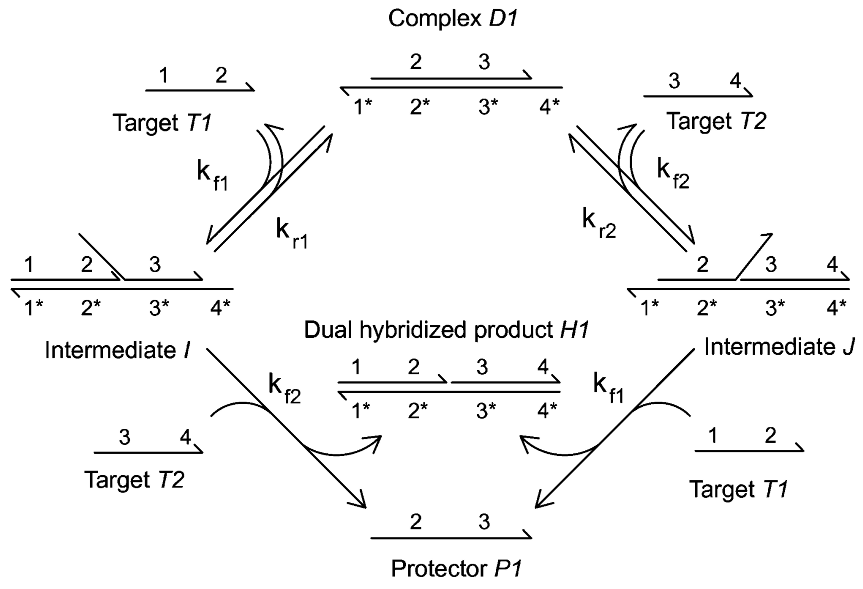
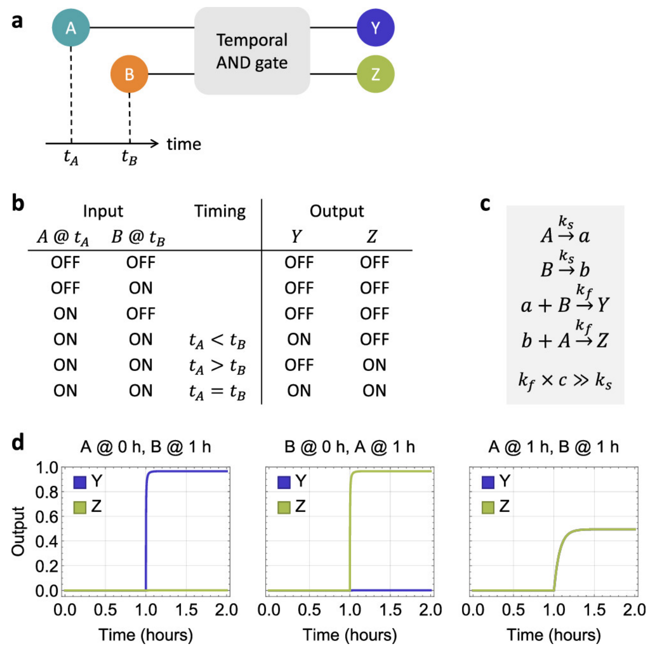


Figure 5.1: Cooperative hybridization mechanism. Two nucleic acid molecules of interest, T1 and T2, cooperatively hybridize to two-stranded complex D1 (composed of upper strand P1 and lower strand L1). T1 and T2 individually bind to D1 in a reversible manner, but their simultaneous binding releases protector strand P1 and dual-hybridized product H1, rendering the reaction irreversible. Adapted from (David Yu Zhang, 2010).



b

Input		Timing	Output	
A @ t_A	B @ t_B		Y	Z
OFF	OFF		OFF	OFF
OFF	ON		OFF	OFF
ON	OFF		OFF	OFF
ON	ON	$t_A < t_B$	ON	OFF
ON	ON	$t_A > t_B$	OFF	ON
ON	ON	$t_A = t_B$	ON	ON

c

$$A \xrightarrow{k_s} a$$
$$B \xrightarrow{k_s} b$$
$$a + B \xrightarrow{k_f} Y$$
$$b + A \xrightarrow{k_f} Z$$
$$k_f \times c \gg k_s$$

Figure 5.2: Concept and chemical reaction network implementation of temporal logic circuits. (a) Abstract circuit diagram, (b) truth table, (c) chemical reaction network implementation, and (d) simulations of a two-input temporal AND gate. c is the concentration of input signals A and B. Simulations of output signals Y and Z are shown as relative concentrations to c over time, where c = 100 nM, $ks = 0.002/s$, and $kf = 210^6/M/s$. From (Lapteva, Sarraf, and Qian, 2022).

5.2 System design

The temporal logic gate that we built has 2 inputs, A and B, that each contain a logic value, ON or OFF, and relative timing information, t_A and t_B. The truth table has six possible outputs, as opposed to four in a regular two-input AND gate, as shown in Figure 5.2b. The additional entries in the logic table correspond to the two additional ON states of the circuit that indicate relative timing information. In Figure 5.2c, the chemical reactions that describe the functioning of the system are shown. First, inputs A and B are converted into memory species, a and b, via a slow reaction at rate k_s. The memories then react with the second-arriving signal to produce an output via a fast reaction at rate k_f. The differences in reaction speed are necessary so that when the second input arrives, it reacts with the memory species rather than being converted into a memory species itself. No output can be produced if neither or only one of the input signals has arrived. Simulations of the circuit behavior are shown in Figure 5.2d.

5.3 Methods

Sample preparation

DNA oligonucleotides were synthesized by Integrated DNA Technologies (IDT). Strands without modifications wereordered with standard desalting, LabReady at 100 μM in IDTE buffer at pH 8.0. Strands with a fluorophore orquencher were ordered with high-performance liquid chromatography (HPLC) purification. Annealing of complexes was performed in an Eppendorf thermocycler. The samples were cooled from 90°C to 20°C over the course of 90 minutes. At 90°C, all strands should be single stranded with no secondary structures. As they cool down, they should preferentially form the designed complexes. The final buffer condition of all complexes was 1×TE with 12.5 mM Mg^{2+}. Reporter complexes were not purified after annealing, but gate complexes were purified using polyacrylamide gelelectrophoresis (PAGE). A 12% PAGE gel was run for 6 hours at 150 V in 1xTAE with 12.5 mM Mg2+, with the exposed wire in the upper chamber of the gel box being cleared of salt every 1.5 hours. The relevant bands were incubated in 1×TE buffer with 12.5 mM Mg2+ for at least 24 hours. The NanoDrop (ThermoFisher) was used to quantify the concentrations of the complexes after purification. DNA absorbs UV light at 260 nm, so the measurement at that wavelength was used to calculate the concentration, normalized by the specific extinction coefficient of that species.

Fluorescence kinetics experiments

Experiments were run with species concentrations specified in figure captions. Master mixes containing all shared circuit components were prepared in tubes and input mixes were added to a clear- and flat-bottomed 96-well plate.Master mixes were then added to the plate and mixed thoroughly. In instances where input signals were added at different times, the second signal was added after one hour incubation at room temperature. The plate was then centrifuged and inserted into a microplate reader (Synergy H1, Biotek). Fluorescent readings were taken from the bottom of the plate at room temperature (approximately 22◦C). The excitation/emission wavelengths were 496/525 nm for ATTO488 and 598/629 nm for ATTO590.

5.4 Experimental implementation

The molecular implementation of these reactions is shown in Figure 5.3a. Input A irreversibly triggers seesaw gate Gate A. The output, species a, is one input to the cooperative hybridization gate Gate aB. The other reactant of the cooperative hybridization gate is B. When both a and B are present and cooperatively hybridize

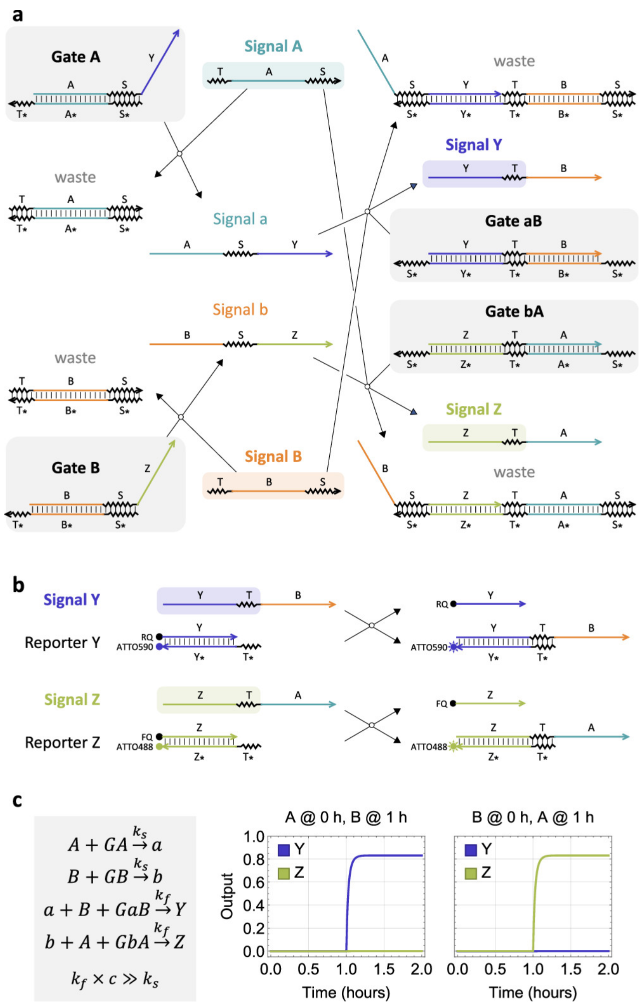
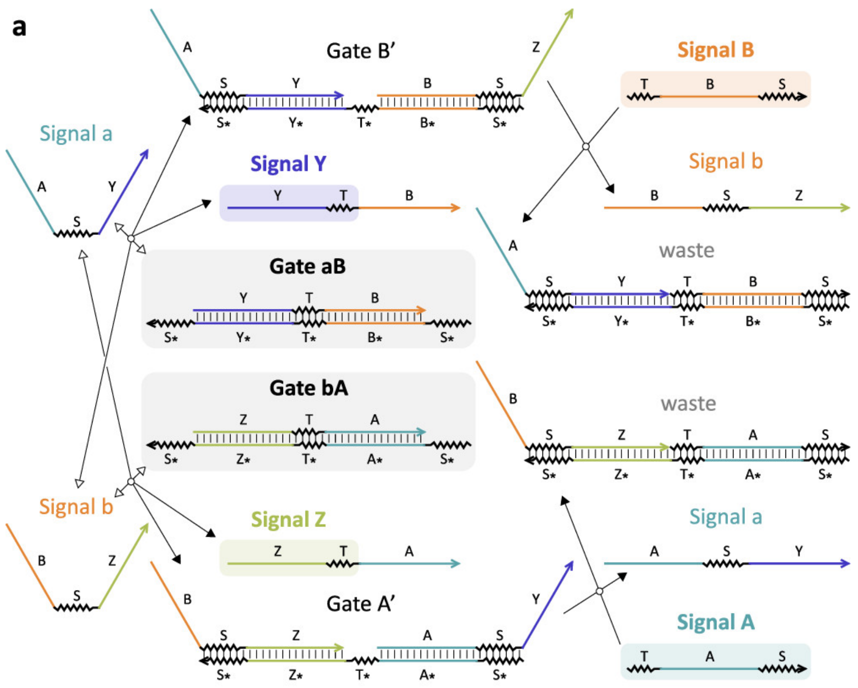


Figure 5.3: DNA strand-displacement implementation of a two-input temporal AND gate. (a) Reaction pathways. (b) Reporting mechanism. Zigzagged and straight lines indicate short toehold and long branch migration domains, respectively. Asterisks in domain names indicate sequence complementarity. (c) Simulations. c is the concentration of input signals A and B. Output signals Y and Z are shown as relative concentrations to c over time, where c = 100 nM, $k_s = 10^5/M/$, and $k_f = 210^{13}/M^2/s$. Gates and reporters were in 20% and 50% excess compared to inputs, respectively. Adapted from (Lapteva, Sarraf, and Qian, 2022).

to Gate aB, the output Signal Y will be released and can react with the downstream reporter. The reporting reaction is shown in Figure 5.3b. The same logic holds for the other pathway, if B arrives before A. In order to make the cooperative hybridization reaction faster than the initial signal conversion into a memory, the toeholds on the cooperative hybridization gate were extended to 7 nucleotides, while the toeholds on the seesaw gates were kept at 5 nucleotides. Simulation reactions are shown in Figure 5.3c. k_s is the estimated strand displacement rate with a 5-

nucleotide toehold, whereas k_f is the estimated cooperative hybridization rate with a 7-nucleotide toehold.

a



b

$$a + b + GaB \underset{k_{s1}}{\overset{k_{f1}}{\rightleftharpoons}} Y + GB'$$

$$b + a + GbA \underset{k_{s2}}{\overset{k_{f2}}{\rightleftharpoons}} Z + GA'$$

$$A + GA' \xrightarrow{k_{s1}} a$$

$$B + GB' \xrightarrow{k_{s2}} b$$

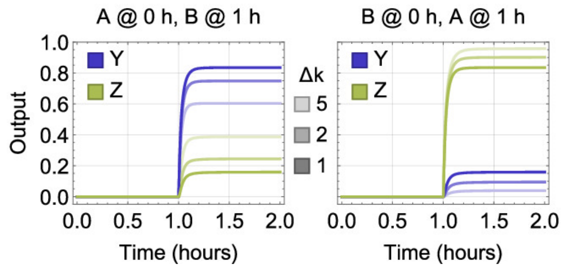


Figure 5.4: Crosstalk between two reaction pathways. (a) Crosstalk reactions. (b) Simulations of the desired reactions (darkest), 2-fold difference in the pairs of rate constants (medium), and 5-fold different in the pairs of rate constants (lightest). Adapted from (Lapteva, Sarraf, and Qian, 2022).

Though these rates were designed such that $k_f >> k_s$, there is still competition between the reaction pathways. The second-arriving input could interact with its upstream gate, e.g. if *Signal B* were to arrive second and interact with *Gate B* to produce b. The small amount of memory strands produced from the second input could then cooperatively hybridize with memory strands from the first input to incorrectly release an output strand, e.g. a and b could bind to *Gate aB* to erroneously produce *Signal Y*, as shown in Figure 5.4a. Simulations of this leak and crosstalk pathway demonstrated an elevated OFF signal and reduced ON signal.

These results are shown in Figure 5.4b. Figure 5.5 shows more detailed results of two types of simulations. In 5.5a, the system is modeled as trimolecular reactions without crosstalk, while in 5.5b they are modeled as trimolecular reactions with crosstalk. With the crosstalk reactions included in the model, there is about 20% leak of the incorrect output.

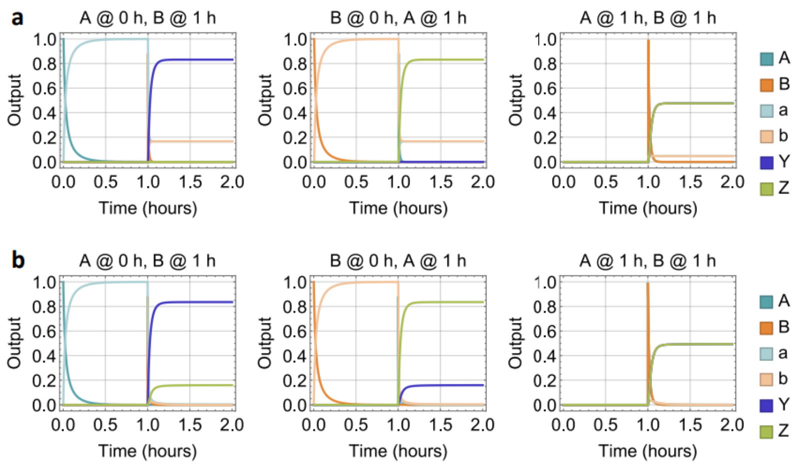
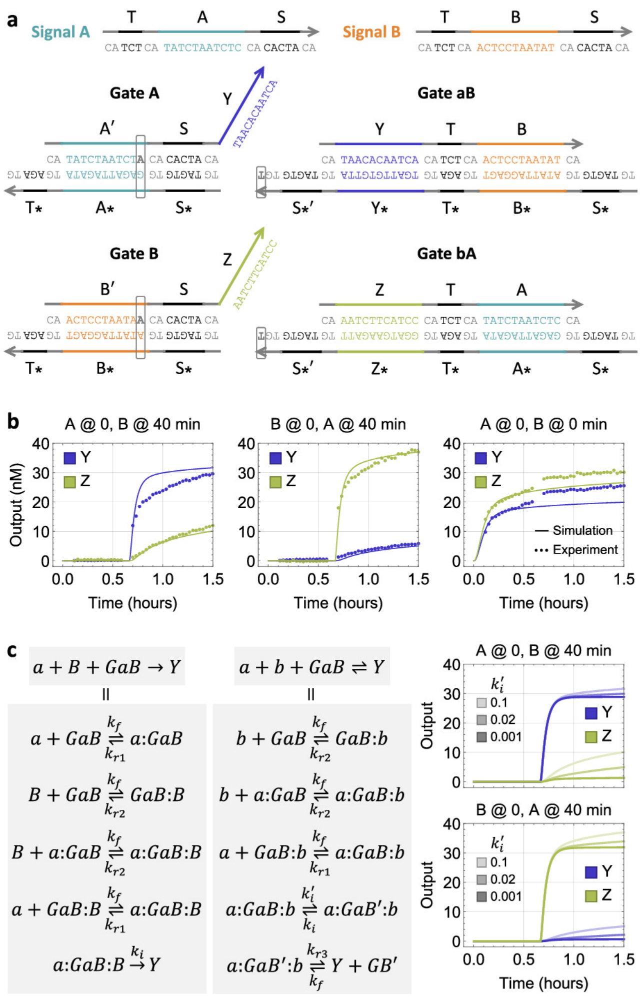


Figure 5.5: Detailed simulation of reaction pathways with and without leak. (a) Cooperative hybridization gates modeled as trimolecular reactions. (b) Cooperative hybridization gates modeled as trimolecular reactions, including crosstalk reactions. Adapted from (Lapteva, Sarraf, and Qian, 2022).

5.5 Debugging the circuit

We saw that crosstalk did indeed affect the circuit functioning in the experimental data, shown in Figure 5.7a. To mitigate the crosstalk while keeping the the circuit implementation as simple as possible, we decided to introduce a mismatch into each memory strand near the 3′ end of the branch migration domain. This should reduce crosstalk by slowing down the reaction of the memories with the cooperative hybridization gate. The position of the mismatch is shown in Figure 5.6a. Studies have shown that if a mismatch is located close to the end of a branch migration domain, it should not have a significant effect on the strand displacement rate, likely because at that position in the reaction the strands are starting to dissociate (Haley et al., 2020). On the other hand, if the mismatch is close to the toehold, it can slow down the kinetics by two orders of magnitude (Machinek et al., 2014).

The data in Figure 5.7a also revealed a rate difference between the two different reaction pathways, in which output Z is produced more quickly than output Y. This



$a + B + GaB \rightarrow Y$

$$a + GaB \underset{k_{r1}}{\overset{k_f}{\rightleftharpoons}} a{:}GaB$$
$$B + GaB \underset{k_{r2}}{\overset{k_f}{\rightleftharpoons}} GaB{:}B$$
$$B + a{:}GaB \underset{k_{r2}}{\overset{k_f}{\rightleftharpoons}} a{:}GaB{:}B$$
$$a + GaB{:}B \underset{k_{r1}}{\overset{k_f}{\rightleftharpoons}} a{:}GaB{:}B$$
$$a{:}GaB{:}B \xrightarrow{k_i} Y$$

$a + b + GaB \rightleftharpoons Y$

$$b + GaB \underset{k_{r2}}{\overset{k_f}{\rightleftharpoons}} GaB{:}b$$
$$b + a{:}GaB \underset{k_{r2}}{\overset{k_f}{\rightleftharpoons}} a{:}GaB{:}b$$
$$a + GaB{:}b \underset{k_{r1}}{\overset{k_f}{\rightleftharpoons}} a{:}GaB{:}b$$
$$a{:}GaB{:}b \underset{k_i}{\overset{k_i'}{\rightleftharpoons}} a{:}GaB'{:}b$$
$$a{:}GaB'{:}b \underset{k_f}{\overset{k_{r3}}{\rightleftharpoons}} Y + GB'$$

Figure 5.6: Characterization of circuit behavior. (a) The position of the mismatch is circled in gray on all Gate molecules. (b) Simulations and fluorescence kinetics data. (c) Simulations with varying branch migration rates. Adapted from (Lapteva, Sarraf, and Qian, 2022).

resulted in a very small separation in the case that *A* arrived before *B*. One principle we use for designing sequences is that CG pairs should have the same spread in the branch migration domain for sequences that we want to have similar branch migration rates. We noticed that the branch migration domain of B had a less even spread than the other domains A, Y, and Z. We therefore altered the branch migration domain of B to more closely match the CG spread of the other three.

We saw that adding the mismatches suppressed the production of undesired output, as shown in Figure 5.7b. Altering the spread of CG pairs in the branch migration domain of B also resulted in more balanced reaction rates, as shown in Figure 5.7c. With the mismatch and the altered CG spread, the separation between the ON and

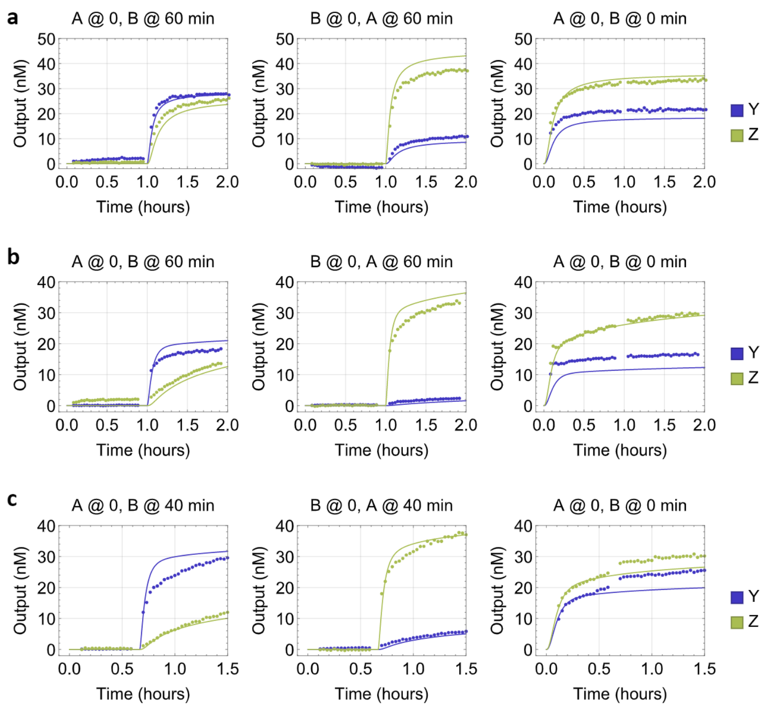


Figure 5.7: Introducing a mismatch into the memory strands. (b) No mismatch. (b) Mismatch in the memory strands. (c) Mismatch in the memory strands and altered domain B to have a more even CG spread. Adapted from (Lapteva, Sarraf, and Qian, 2022).

OFF rates were more similar to each other in the cases of $t_A < t_B$ and $t_B < t_A$.

5.6 A two-input temporal AND gate

To further reduce the likelihood of the memory strand binding to the wrong side of the gate, we decided to shorten the toehold on the 5′ end of the gate that the memory strand could incorrectly bind to. This would also lead to more of the second-arriving input strand being converted to memory, because it would slow down its reaction with the cooperative hybridization gate, and simulations indicated that it would lead to a lowered output signal. However, we thought this tradeoff may still overall be beneficial. The shortened toeholds on those gates are shown in Figure 5.8a.

The experimental results showed that the shortened 5-nucleotide toehold resulted in the best separation between ON and OFF states, as shown in Figure 5.8b, and we were able to successfully implement the two-input temporal AND gate. The results did show a decrease in maximum output signal, as predicted by the simulations. This could be corrected with a signal restoration layer if this module were to be connected to a downstream circuit.

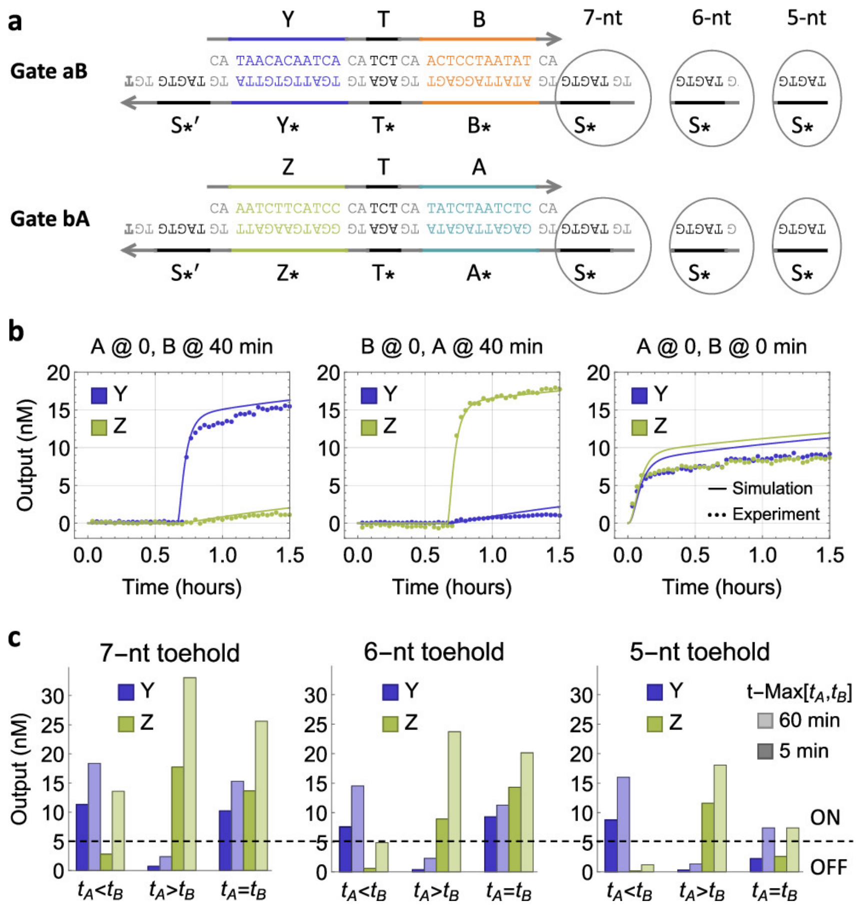


Figure 5.8: Demonstration of the two-input temporal AND gate with varying toehold lengths. (a) Sequence design with varying lengths of the S* toehold on the two cooperative hybridization gates. (b) Simulation and fluorescence kinetics data of the circuit with a 5-nt S* toehold. All gates, reporters, and inputs were at 100, 150, and 90 nM, respectively. (c) Output concentrations in experiments with varying toehold lengths. Darker and lighter bars correspond to output concentrations immediately (within 5 min, when the first data point was collected) and 1 h after the second input has arrived, respectively. Dashed line marks the separation between ON and OFF states. From (Lapteva, Sarraf, and Qian, 2022).

Finally, we investigated the time resolution of the circuit to determine the minimum difference in arrival time between the two signals in order for the circuit to produce the correct the output. Theoretically, the minimum time interval is 100 seconds to allow for the first signal to convert to a memory. Based on simulation, for the ON output to be at least twice the concentration of the OFF output, the minimum difference in arrival time between the two inputs was 5 min. The experimental data showed that ON-OFF separation could be observed with a 1 min time interval, which was smaller than what the theory and simulation predicted.

5.7 Discussion

We have demonstrated the functioning of a DNA strand displacement circuit that uses temporal memory to output information about the order in which two inputs have arrived. In theory, the number of inputs that the circuit detects could be increased by adding more memory species. However, the correct functioning of the circuit would depend on precise balancing of reaction rates that may be practically infeasible. Here we were able to make adjustments that compensated for the rate differences, e.g. adding mutations and shortening one toehold, but with a larger number of molecules it would become increasingly difficult to effectively implement such strategies to balance reaction rates. It may prove simpler and more scalable to explore other methods that have shown promise in recording timing information in molecules, such as extending DNA strands and then sequencing them to read out the event order (Kishi et al., 2018; Cardelli, 2021).

Even so, the implementation of this circuit expands the possibility of being able to use relative timing of signals as a control mechanism for programmed molecular systems. Given how broadly temporal memory is used in biology, having the capability to introduce it into systems that we build would be useful, particularly in in vivo applications in which relative timing of signals triggers biological processes. One could imagine a tile displacement system that reconfigures into one structure for some purpose if signal x were to arrive before signal y, into another structure for a different purpose if signal y were to arrive before signal x, and does not reconfigure at all if one or both of the input signals is missing. A combinatorial number of behaviors could be executed in response to the relative arrival time of input signals.

Chapter 6

CONCLUSION

In this thesis, we have discussed work that ranges from molecular robots to tile-tile interactions to molecular circuits. We have explored a new motif for molecular robotic behavior, track-modifying four-way strand displacement; we have demonstrated that tile displacement can be used to rearrange tile assemblies with complex shapes using tile assemblies with different complex shapes; and we have designed and implemented two new DNA strand-displacement circuits, one that computes a loser-take-all function and one that demonstrates temporal logic. On the surface, these are all quite different types of systems, requiring different strategies and tools, and they are – but one unifying principle that unites these disparate systems is that it is possible to build highly complex systems using one simple building material, DNA.

Even more compelling is the notion that because these diverse systems all share the commonality that they are built out of DNA and use related or overlapping principles and architectures, there is potential for them to be composed together in powerful ways. Molecular robots are particularly adept at performing mechanical tasks, but they are not yet as sophisticated at information-processing as DNA circuits are. Tile displacement reactions are a robust mechanism for reconfiguring 2D and 3D structures, but their input signals are other tiles, arrays, or assemblies, which are less robust than single DNA strands that could also be RNA or proteins. And DNA circuits do not have the capability to perform mechanical tasks or structurally reconfigure. Thus, we imagine a molecular computer that elegantly and powerfully exploits these various advantages of molecular systems: for example, a temporal logic gate that process information from the environment and whose output triggers a tile displacement reaction, which reconfigures the local environment of a molecular robot that is located on top of the tiles. Thus, relative timing information from environmental signals could be used to direct the behavior of a molecular robot via its changing local environment. For a maze-solving robot, this could mean updating the goal location that it needs to reach. In that case, being able to use more complex tile displacement assemblies means that the design space for the kinds of local environments, such as mazes, that could be modulated in this way is greatly expanded. In addition to a temporal logic gate, in a noisy molecular environment, in which many signals might have similar concentrations, a function like loser-take-all would be particularly useful to process information as well. It would allow for the identification of a signal that is out of alignment with the expected value, for example in a disease state in which one biomarker is particularly low. These are just some examples of the kinds of systems that could be built – there are any number of other possibilities, such as connecting a DNA circuit directly to a molecular robot. An output from a molecular robot could, together with a pre-existing environmental signal, trigger a temporal logic gate output that then triggers further downstream modules.

Of course, a key component to building these kinds of integrated systems is the development of robust interfaces between them. There is a natural interface between DNA circuits and molecular robots, in which a circuit could output a strand that triggers the robot to start moving, or the robot reaching its goal could release a strand that acts as an input to a DNA circuit. There is also a natural alignment between molecular robots and tile displacement systems, in that many of the DNA robots that have been developed already walk on the surface of origami tiles. However, the main missing piece is the interface to and from single strands and tile displacement assemblies. It is possible to imagine a type of interface in which a single strand reacts with a tile edge to reveal a toehold, or two tiles coming together results in a single strand being released. In fact, we have begun to explore this type of interface alongside other researchers. But it remains to be seen whether this will prove to be a robust mechanism for composing tile displacement systems together with circuits and robots.

We feel confident that it will take on the order of years, rather than decades, to realize these kinds of integrated molecular systems. We have all the tools and strategies that we need to develop the interfacing technology that is needed. We predict that as this happens, and it becomes more of a reality that we can write molecular programs that perform complex information processing and then affect change on the physical world, it will also become more apparent what the killer applications of such systems are. Perhaps it will be *in vivo*, responding to complex biological signals and reconfiguring cellular structures that are tethered to origami tiles. Perhaps it will be *in vitro*, identifying harmful chemicals in our food and wellness products and marking pathways to them that allow worker robots to remove them or reconfigure their chemical structures to no longer be a danger to humans. Perhaps it will be something entirely different that we cannot even yet imagine, like aluminum finding its killer application in airplanes 30 years after it became possible to mass produce it. It is more likely that there will be many killer applications, and they will span a variety of domains.

Whatever these technologies will look like or be used for in the future, this researcher is convinced of one notion. Integrated molecular systems will be the route by which the beauty and utility of the hybrid software-hardware nature of molecular programming will reach its fullest potential.

BIBLIOGRAPHY

Adleman, Leonard M. (1994). "Molecular Computation of Solutions to Combinatorial Problems". In: *Science* 266, pp. 1021–1024.

Bennett, C. H. (1973). "Logical Reversibility of Computation". In: *IBM J. Res. Dev.* 17, pp. 525–532.

– (1982). "The thermodynamics of computation - a review". In: *Int. J. Theor. Phys.* 21, pp. 905–940.

Binnig, G., C. F. Quate, and Ch. Gerber (1986). "Atomic Force Microscope". In: *Phys. Rev. Let.* 56, pp. 930–934.

Bray, Dennis (2011). *Wetware: A Computer in Every Living Cell*. Yale University Press. ISBN: 9780300167849.

Cardelli, Luca (2021). "Sequenceable Event Recorders". In: *arXiv*.

Chao, Jie et al. (2019). "Solving mazes with single-molecule DNA navigators". In: *Nature Mat.* 18, pp. 273–279.

Chatterjee, Gourab et al. (2017). "A spatially localized architecture for fast and modular DNA computing". In: *Nature Nanotech.* 12, pp. 920–927.

Cherry, Kevin M. and Lulu Qian (2018). "A molecular multi-gene classifier for disease diagnostics". In: *Nature Chem.* 10, pp. 746–754.

Clamons, Samuel, Lulu Qian, and Erik Winfree (2020). "Programming and simulating chemical reaction networks on a surface". In: *J. R. Soc. Interface* 17.

Dabby, Nadine L. (2013). "Synthetic Molecular Machines for Active Self-Assembly: Prototype Algorithms, Designs, and Experimental Study". PhD thesis. California Institute of Technology.

Dannenberg, Fritz et al. (2015). "DNA walker circuits: computational potential, design, and verification". In: *Natural Computing* 14, pp. 195–211.

Doty, David et al. (2017). "Thermodynamic Binding Networks". In: *Lecture Notes in Computer Science*. Ed. by R. Brijder and Lulu Qian. Vol. 10467, pp. 249–266.

Edwards, Cristofer J., Todd B. Alder, and Gary J. Rose (2002). "Auditory midbrain neurons that count". In: *Nature Neuro.* 5.

Franks, Nigel R. et al. (2015). "How ants use quorum sensing to estimate the average quality of a fluctuating resource". In: *Scientific Reports* 5.

Friedland, ARI E. et al. (2009). "Synthetic Gene Networks That Count". In: *Science* 324, pp. 1199–1202.

Fu, Tsu-Ju and Nadrian C. Seeman (1993). "DNA Double-Crossover Molecules". In: *Biochem.* 32, pp. 3211–3220.

Gu, Hongzhou et al. (2010). “A proximity-based programmable DNA nanoscale assembly line”. In: *Nature Letters* 465, pp. 202–206.

Haddad, Rafi et al. (2013). “Olfactory cortical neurons read out a relative time code in the olfactory bulb”. In: *Nature Neuro.* 16, pp. 949–957.

Haley, Natalie E. C. et al. (2020). “Design of hidden thermodynamic driving for non-equilibrium systems via mismatch elimination during DNA strand displacement”. In: *Nature Comm.* 11.

Hays, Franklin A., Jeffrey Watson, and P. Shing Ho (2003). “Caution! DNA Crossing: Crystal Structures of Holliday Junctions”. In: *J. Biol. Chem.* 278, pp. 49663–49666.

Ho, P. Shing and Brandt F. Eichman (2001). “The crystal structures of DNA Holliday junctions”. In: *Curr. Op. Struc. Biol.* 11, pp. 302–308.

Holliday, Robin (1964). “A mechanism for gene conversion in fungi”. In: *Genet. Res.* 5, pp. 282–304.

Hopfield, J. J. (1974). “Kinetic Proofreading: A New Mechanism for Reducing Errors in Biosynthetic Processes Requiring High Specificity”. In: *Proc. Nat. Acad. Sci.* 71, pp. 4135–4139.

Hsiao, Victoria et al. (2016). “A population-based temporal logic gate for timing and recording chemical events”. In: *Mol. Sys. Biol.* 12.

Kishi, Jocelyn Y. et al. (2018). “Programmable autonomous synthesis of single-stranded DNA”. In: *Nature Chem.* 10, pp. 155–164.

Knudsen, Jakob Bach et al. (2015). “Routing of individual polymers in designed patterns”. In: *Nature Nanotech.* 10, pp. 892–898.

Kopperger, Enzo et al. (2018). “A self-assembled nanoscale robotic arm controlled by electric fields”. In: *Science* 359, pp. 296–301.

Lakin, Matthew R. and Darko Stefanovic (2016). “Supervised Learning in Adaptive DNA Strand Displacement Networks”. In: *ACS Synth. Biol.* 5, pp. 885–897.

– (2017). “Towards Temporal Logic Computation Using DNA Strand Displacement Reactions”. In: *Lecture Notes in Computer Science*. Ed. by M. Patitz and M. Stannett. Vol. 10240.

Lapteva, Anna P., Namita Sarraf, and Lulu Qian (2022). “DNA Strand Displacement Temporal Logic Circuits”. In: *J. Am. Chem. Soc.* 144, pp. 12443–12449.

Li, Jieming et al. (2018). “Exploring the speed limit of toehold exchange with a cartwheeling DNA acrobat”. In: *Nature Nanotech* 13, pp. 723–729.

Li, Xiaojun et al. (1996). “Antiparallel DNA Double Crossover Molecules As Components for Nanoconstruction”. In: *J. Am. Chem. Soc.* 118, pp. 6131–6140.

Lilley, David M. J. (2000). “Structures of helical junctions in nucleic acids”. In: *Q. Rev. Biophys.* 33, pp. 109–159.

Lin, Yihan et al. (2015). “Combinatorial gene regulation by modulation of relative pulse timing”. In: *Nature* 527, pp. 54–58.

Liu, Chanjuan et al. (2020). “Cross-Inhibitor: a time-sensitive molecular circuit based on DNA strand displacement”. In: *Nuc. Acids Res.* 48, pp. 10691–10701.

Liu, Qinghua et al. (2000). “DNA computing on surfaces”. In: *Nature* 403, pp. 175–179.

Lopez, Randolph, Ruofan Wang, and Georg Seelig (2018). “Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks”. In: *Nature* 559, pp. 370–376.

Lund, Kyle et al. (2010). “Molecular robots guided by prescriptive landscapes”. In: *Nature* 465, pp. 206–210.

Machinek, Robert R. F. et al. (2014). “Programmable energy landscapes for kinetic control of DNA strand displacement”. In: *Nature Comm.* 5.

Mao, Chengde et al. (2000). “Logical computation using algorithmic self-assembly of DNA triple-crossover molecules”. In: *Nature* 407, pp. 493–496.

Muscat, Richard A., Jonathan Bath, and Andrew J. Turberfield (2011). “A Programmable Molecular Robot”. In: *Nano Lett.* 11, pp. 982–987.

O’Brien, Jackson and Arvind Murugan (2019). “Temporal Pattern Recognition through Analog Molecular Computation”. In: *ACS Synth. Biol.* 8, pp. 826–832.

Omabegho, Tosan, Ruojie Sha, and Nadrian C. Seeman (2009). “A Bipedal Brownian Motor With Coordinated Legs”. In: *Science* 324, pp. 67–71.

Panyutin, Igor G. and Peggy Hsieh (1993). “Formation of a single base mismatch impedes spontaneous DNA branch migration”. In: *J. Mol. Bio.* 230, pp. 413–424.

– (1994). “The kinetics of spontaneous DNA branch migration”. In: *Proc. Natl. Acad. Sci.* 91, pp. 2021–2025.

Petersen, Philip, Grigory Tikhomirov, and Lulu Qian (2014). “Poster: A maze-solving DNA robot for prototyping complex nanomechanical tasks performed using simple molecular components”. In: *DNA 20, Kyoto, Japan*.

– (2018). “Information-based autonomous reconfiguration in systems of interacting DNA nanostructures”. In: *Nature Comm.* 9.

Qian, Lulu and Erik Winfree (2011). “Scaling up digital circuit computation with DNA strand displacement cascades”. In: *Science* 332, pp. 1196–1201.

Qian, Lulu, Erik Winfree, and Jehoshua Bruck (2011). “Neural network computation with DNA strand displacement cascades”. In: *Nature* 475, pp. 368–372.

Redding, Charles M. et al. (1977). “Uptake of homologous single-stranded fragments by superhelical DNA: IV. Branch migration”. In: *J. Mol. Bio.* 116, pp. 825–839.

Rodriguez, Kellen R., Namita Sarraf, and Lulu Qian (2021). “A Loser-Take-All DNA Circuit”. In: *ACS Synth. Biol.* 10, pp. 2878–2885.

Rothemund, Paul W. K. (2006). “Folding DNA to create nanoscale shapes and patterns”. In: *Nature* 440, pp. 297–302.

Rothemund, Paul W. K., Nick Papadakis, and Erik Winfree (2004). “Algorithmic Self-Assembly of DNA Sierpinski Triangles”. In: *PLoS Biol* 2.

Seelig, Georg et al. (2006). “Enzyme-Free Nucleic Acid Logic Circuits”. In: *Science* 314, pp. 1585–1588.

Seeman, Nadrian C. (1998). “Nucleic Acid Nanostructures and Topology”. In: *Angew. Chem. Int. Ed.* 37, pp. 3220–3238.

Seeman, Nadrian C. and Oleg Gang (2017). “Three-dimensional molecular and nanoparticle crystallization by DNA nanotechnology”. In: *MRS Bulletin* 42, pp. 904–912.

Shin, Jong-Shik and Niles A. Pierce (2004). “A Synthetic DNA Walker for Molecular Transport”. In: *J. Am. Chem. Soc.* 126, pp. 10834–10835.

Simmel, Friedrich C., Bernard Yurke, and Hari R. Singh (2019). “Principles and Applications of Nucleic Acid Strand Displacement Reactions”. In: *Chem. Rev.* 119, pp. 6326–6369.

Soloveichik, David (2009). *CRNSimulator*. `http://users.ece.utexas.edu/ÌČ~soloveichik/crnsimulator.html/`.

Srinivas, Niranjan et al. (2013). “On the biophysics and kinetics of toehold-mediated DNA strand displacement”. In: *Nuc. Acids Res.* 41, pp. 10641–10658.

Stefanovic, Darko (2012). “Maze Exploration with Molecular-Scale Walkers”. In: *TPNC 2012: Theory and Practice of Natural Computing*.

Thubagere, Anupama J. et al. (2017). “A cargo-sorting DNA robot”. In: *Science* 357.

Tikhomirov, Grigory, Philip Petersen, and Lulu Qian (2017a). “Fractal assembly of micrometre-scale DNA origami arrays with arbitrary patterns”. In: *Nature* 552, pp. 67–71.

– (2017b). “Programmable disorder in random DNA tilings”. In: *Nature Nanotech.* 12, pp. 251–259.

– (2018). “Triangular DNA tilings”. In: *J. Am. Chem. Soc.* 140, pp. 17361–17364.

Wang, Hao (1963). In: *Proceedings of a Symposium in the Mathematical Theory of Automata*.

Wickham, Shelley F. J. et al. (2012). “A DNA-based molecular motor that can navigate a network of tracks”. In: *Nature Nanotech.* 7, pp. 169–173.

Winfree, Erik (1998). “Algorithmic Self-Assembly of DNA”. PhD thesis. California Institute of Technology.

Winfree, Erik, Furong Liu, et al. (1998). “Design and self-assembly of two-dimensional DNA crystals”. In: *Nature* 394, pp. 539–544.

Winfree, Erik and Lulu Qian (2022). “Two-dimensional tile displacement can simulate cellular automata”. In: *In preparation*.

Yin, Peng et al. (2004). “A Unidirectional DNA Walker That Moves Autonomously Along a Track”. In: *Angew. Chem. Int. Ed.* 43, pp. 4906–4911.

Yurke, Bernard and Allen P. Mills Jr. (2003). “Using DNA to Power Nanostructures”. In: *Genet. Program Evolvable Mach.* 4, pp. 111–122.

Yurke, Bernard, Andrew J. Turberfield, et al. (2000). “A DNA-fuelled molecular machine made of DNA”. In: *Nature* 406, pp. 605–608.

Zhang, David Y. and Erik Winfree (2009). “Control of DNA Strand Displacement Kinetics Using Toehold Exchange”. In: *J. Am. Chem. Soc.* 131, pp. 17303–17314.

Zhang, David Yu (2010). “Cooperative Hybridization of Oligonucleotides”. In: *J. Am. Chem. Soc.* 133, pp. 1077–1086.

Zhang, David Yu et al. (2007). “Engineering Entropy-Driven Reactions and Networks Catalyzed by DNA”. In: *Science* 318, pp. 1121–1125.

Appendix A

ADDITIONAL DATA

Appendix B

THE ART OF MOLECULAR PROGRAMMING

The Art of Molecular Programming is a passion project that I started with Dominic Scalise, who was a postdoc in our lab at the time and is now a professor at Washington State University. The inspiration for the general ideas that would morph into this project struck me first at DNA 25, held at the University of Washington in Seattle. I began to think about how we cultivated community in our field outside of our two main conferences, DNA and fNANO. How did we provide a basis on which researchers could discuss and agree upon overarching goals, challenges, messaging, and ethical issues, as they seemed to do in other fields? How did we cultivate an identity for our field, one that we felt proud of and that could garner excitement in a lay audience the same way a term like "quantum computing" does?

For a decade, the field of molecular programming had been supported in its growth by a series of large-scale grants from the NSF called the Molecular Programming Project (MPP). MPP had served as a gravitation point for an increasing number of investigators, and having come to an end, left a bit of a vacuum in its wake. The answer came to me that after MPP, it as crucial to start to develop a new framework with which to guide and grow and field.

It seemed that it would be up to us to develop the infrastructure upon which we could foster the community collaboration that we wanted to see. In discussing this undertaking post-conference, Dominic suggested that an introductory textbook would be a good first step in establishing our identity as a field. It was something we had all sorely missed when we first embarked on our molecular programming journeys, sorting through a growing body of papers to understand the accepted dogmas of the field. It has matured to the point that there are well-understood and well-accepted principles to gather, weave together, and encode.

Thus, the Art of Molecular Programming Project was born. We envisioned it as a community-driven effort to write a textbook that would serve as an introduction and inspiration for anyone curious about or entering the field; a reference for those already working in molecular programming; an account of what we have accomplished as a field, where we have been; and a blueprint for where we might be going.

Our goal was to recruit graduate students, postdocs, and faculty from all corners of what we might consider molecular programming to work on the book. Not only would that give us the widest expertise on a range of topics, but we hoped that the project itself would become a vibrant community for enthusiastic molecular programmers (which it has!).

As co-Executive Editors, Dominic and I first grew our team to six Executive Board members. We also brought on a Faculty Advisory Board to provide feedback on the scope of the book as well as get buy-in from more veteran researchers who are the real experts on the foundational topics of the field. In honing the table of contents with this initial team, we settled on three sections of the book: Structures and Self-Assembly, Circuits and Information Processing, and Interfaces.

For each of these sections, we brought on a Topic Lead and 5-7 Topic Editors. They are responsible for shaping the content of their sections, recruiting authors, and editing their authors' work before incorporating it into the larger text. It was a particular joy to solicit applications for these positions and see so much enthusiasm and expertise from all over the world, in all different areas of the field.

That we have contributors from across almost every continent is something I attribute to the project coming to fruition during the Covid-19 pandemic. Meeting over Zoom - sometimes at hours that are odd for one or more of the participants - worked seamlessly into our lifestyles in a way that might have seemed impossible before March 2020. And without the social starvation that we all faced for so long, our editors may not have been as eager to spend so much time talking to their colleagues through a screen. The silver lining of a dark time!

Because a major goal of this project is to bring the community together, it was important to us that the book be accessible to the widest possible audience. We signed a publishing deal with World Scientific that would allow us to keep copyright of our content as well as publish a free E-book one year after the initial publication date.

At the moment of writing this thesis, we have written work from authors that covers about 50% of the sections in our Table of Contents. Another 30% of the sections have outlines that are currently being turned into drafts. By this time next year, we should have a first edition of the book near to being published.

In particular, I want to thank the initial team that joined Dominic and me on this crazy ride. William Poole, Hannah Earley, Jacob Majikes, Anastasia Ershova, and Elisa Franco - thank you for believing in the project and helping the vision grow bigger and brighter day by day. It has been so rewarding to see each of us pour our hearts into this book, and I am so excited to see the final product come together.

Appendix C

DNA SEQUENCES

T C01 R01	CCCCGGGCGATGGCCCACTAGAAAAACCAACGGGGTCCCC
T C01 R02	CCCCAACGGTACGCCAGAATAGGGATTTTAGACAGGCCCC
T C01 R03	CCCCAAGAATACGTGGCACATCTGACCTATGATACAGGAGTGTA
T C01 R04	CCCCTTGAGGATTTAGAAGTTCAATAGATAATACATCCCC
T C01 R05	CCCCATAACGGATTCGCCTGTACATCGGCCGTTCCAGTAAGCG
T C01 R06	CCCCTAGGTTGGGTTATATATTTTTAACCTCCGGCTCCCC
T C01 R07	CCCCGTACCGACAAAAGGTAAATAAGAGAGCCAGAATGGAAAGC
T C01 R08	CCCCCGGGAGGTTTTGAAGCCGAACCTCCCGACTTGCCCC
T C01 R09	CCCCAACAATGAAATAGCAAAATAATAATGATATTCACAAACAA
T C01 R10	CCCCGGTGAATTATCACCGTGAAATTATTCATTAAACCCC
T C02 R01	AAAGGGCCGTGAACCATCACCCCAGGAGGC
T C02 R02	CGATTAACCTGAGAAGTGTTTTCAGAGATA
T C02 R03	GAACCCTGACAATATTTTTGAAAATAGATT
T C02 R04	AGAGCCGATTAGACTTTACAAAAGTAACAG
T C02 R05	TACCTTTATTGCTTTGAATACCGTCTGAGA
T C02 R06	GACTACCACTATATGTAAATGCCATTTTCG
T C02 R07	AGCCAGTAAGTAATTCTGTCCACGCGAGGC
T C02 R08	GTTTTAGCTTAAATCAAGATTAAATTGAGT
T C02 R09	TAAGCCCTAGCTATCTTACCGAAGGTAAAT
T C02 R10	ATTGACGCACCGACTTGAGCCACACCCTCA
T C02 R11	GAGCCGCCACCAGAAAGGAGGTTGAGGCAG
T C03 R01	TTTTTTGGTTAAAGAATTCGGTCGCCCC
T C03 R02	GTGAGGCCCAGAGCGGGAGCTAAAAAATCAAG
T C03 R03	AGTCTTTAGGGACATTCAACAACCCCTCATAGTTAGCGTACAATAGGA
T C03 R04	CAACTCGTAGGAGCACTAACAACTTGGCTATT
T C03 R05	AAATCGCGCGTCAGATAGCTTGATCGTCTTTCCAGACGCCACCACC
T C03 R06	ATCCAATCTTTATCAAAATCATAGAAGTTACA
T C03 R07	CAATAAACGTAATTTAATCAGCTTCTGTATGGGATTTTGCAACCGCCA
T C03 R08	TTTGCACCTCCGGTATTCTAAGAAGACGACGA
T C03 R09	TTAAGAAAAGAGATAACTCCAAAAAGTTTCAGCGGAGTGATACTCAGG
T C03 R10	TTAGAGCCCCGATTGAGGGAGGGAAGCCCTTT
T C04 R01	GTCCACTAGGTCGAGGTGCCGTAACTTTCCTC
T C04 R02	GTTAGAATACCGAGTAAAAGAGTCACGACCAG

T C04 R03	TAATAAAAATGCGCGAACTGATAGTATCTAAA
T C04 R04	ATATCTTTATTAAATCCTTTGCCCAGATTTTC
T C04 R05	AGGTTTAACAGAGGCGAATTATTCAAGAGTCA
T C04 R06	ATAGTGAAGCAAGACAAAGAACGCTTAACAAC
T C04 R07	GCCAACATAACATGTTCAGCTAATAGATATAG
T C04 R08	AAGGCTTACAGCTACAATTTTATCTGAGCGCT
T C04 R09	AATATCAGAGTAAGCAGATAGCCGAAAGGGCG
T C04 R10	ACATTCAAAGCAAAATCACCAGTAGCCACCCT
T C04 R11	CAGAACCGCCACCCTCTTTTTCACTAAAGGAATTGCGACCCC
T C05 R01	AATCGGAATGTTGTTCATTACAGGCCCC
T C05 R02	CACGCAAAAGCACGTATAACGTGAGCACTA
T C05 R03	ACATCGCCAGATTCACCGTTAATAGAAAGAGGACAGATGATCCGCGAC
T C05 R04	ATTAATTTGGAATTGAGGAAGGTCCCTAAA
T C05 R05	TTACCTGAATAAAGAATACCAGTGCGCATAGGCTGGCTTTGTATCA
T C05 R06	CTTTTTCAGATTAAGACGCTGAGATTTCAA
T C05 R07	GCGCCTGTTGAGAATCTACCTTATAATCTTGACAAGAACCATACCAAG
T C05 R08	TTACCAACAATAGCAAGCAAATCGCAGAAC
T C05 R09	TTACCAGAAAGTCAGATGGTTTAAAAATCAACGTAACAAACTAAAACA
T C05 R10	TACCATTACCAGCGCCAAAGACAAACAAAG
T C06 R01	GGGTTGAGCCCTAAAGGGAGCCCCTATGGTTG
T C06 R02	CTTTGACGTTAACCGTTGTAGCAAGATTATTT
T C06 R03	ACATTGGCATTAAAAATACCGAACAAATCAAC
T C06 R04	AGTTGAAATAAAAGTTTGAGTAACTGCACGTA
T C06 R05	AAACAGAAGCAAAAGAAGATGATGACATAGCG
T C06 R06	ATAGCTTAAATATATTTTAGTTAAAACAGTAG
T C06 R07	GGCTTAATTTATCAACAATAGATAAATCATTA
T C06 R08	CCGCGCCCGCTAACGAGCGTCTTTCTGAACAC
T C06 R09	CCTGAACAAGGAAACCGAGGAAACAAATTCAT
T C06 R10	ATGGTTTAGCAAGGCCGGAAACGTAGCCACCA
T C06 R11	CCGGAACCGCCTCCCTCACCAGAATAAGGCTTGCCCTGCCCC
T C07 R01	AGCTTGACAAATCAAATTTGGGGCCCCC
T C07 R02	TGATTAGTGCTACAGGGCGCGTACCGATTTAG
T C07 R03	AGCAGAAGTCAATCGTATGGTCAATCAAATATCGCGTTTTTCAGGTCT
T C07 R04	TTTGCGGATATCTGGTCAGTTGGCGAACCACC
T C07 R05	ATCAAGAACCATATCAATTTAGTCGAACCAGACCGGAATCAGAAAA
T C07 R06	TCTGACCTCTTAGAATCCTTGAAAAAACAAAC
T C07 R07	ACAAGAAATATAAAGCAACAGTTGCAGGATTAGAGAGTACTCATTGAA
T C07 R08	TAATTTGCTTATTTTCATCGTAGGAGTCCTGA

T C07 R09	AACGGAATTAGACGGGAAGTACGGTTGATAAGAGGTCATTGCGTCCAA
T C07 R10	AAACCATCTGTCACAATCAATAGAGCAATAAT
T C08 R01	CCCTTATGGGGAAAGCCGGCGACGCGCTTA
T C08 R02	ATGCGCCAATAACATCACTTGCCCTACATT
T C08 R03	TTGACGCATAAAACAGAGGTGACAAACCCT
T C08 R04	CAATCAAACAAAGAAACCACCAAAGGGTTA
T C08 R05	GAACCTAAACAAAATTAATTACATTAATTA
T C08 R06	ATTTTCCAAATTTAATGGTTTGACAAATTC
T C08 R07	TTACCAGAATAATATCCCATCCACAAGCAA
T C08 R08	GCCGTTTCAGTTACAAAATAAAAACAGGGA
T C08 R09	AGCGCATACCCAAAAGAACTGGGGAATAAG
T C08 R10	TTTATTTGATAGCAGCACCGTAATCTTTTC
T C08 R11	ATAATCAAAATCACCTGTAGCTAGCTTAATTGCTGAACCCC
T C09 R01	AGAAAGGATGGTGGTTAACAAACGCCCC
T C09 R02	AAGAACTCTAACCACCACACCCGCACGTGGCG
T C09 R03	GTATTAACAACGCTCAGCGAGTAAGTCATTGCCTGAGAGTATGATATT
T C09 R04	GGAATTATGCTGAACCTCAAATATGGCGGTCA
T C09 R05	ATTTCATTTACTTCTGTGTAGCCAATCGATGAACGGTAAAAGGCCG
T C09 R06	ACCGTGTGTCTGTAAATCGTCGCTATTTAACA
T C09 R07	GAGCATGTGTATCATACGCCATCAGTCAATCATATGTACCGTAATGTG
T C09 R08	TTATTTATACCGCACTCATCGAGATAATTTAC
T C09 R09	AGACTCCTGAGAGAATTTAAATCAAAAAGCCCCAAAAACAAACCCTCA
T C09 R10	CGACAGAAAACGCAAAGACACCACCATGATTA
T C10 R01	CTGTTTGAAGGGAAGAAAGCGAAAGCGGTCAC
T C10 R02	GCTGCGCGAAACTATCGGCCTTGCCCATTGCA
T C10 R03	ACAGGAAAACCGCCTGCAACAGTGTCTAAAGC
T C10 R04	ATCACCTTCATCATATTCCTGATTCTGATTGT
T C10 R05	TTGGATTATGAATTACCTTTTTTAAGTGAATA
T C10 R06	ACCTTGCTATAAATAAGGCGTTAAAGAAAAAG
T C10 R07	CCTGTTTAAGAAACCAATCAATAAGGGTATTA
T C10 R08	AACCAAGTCCCAATCCAAATAAGAAATAGCAG
T C10 R09	CCTTTACATATTACGCAGTATGTTGGCAACAT
T C10 R10	ATAAAAGATCAAGTTTGCCTTTAGTTTCGGTC
T C10 R11	ATAGCCCCCTTATTAGATATTTTAAATATTTAAATTGTCCCC
T C11 R01	CCCCGTTTGCCCGCAGCAAGCCCC
T C11 R02	CCCCGCGCTGGCAAGTGTAGGAGCGGGCGCTAGGCCCC
T C11 R03	CCCCCAATATTATTGCCCTTTCGTAATCATGGTCATGTTGTAAA
T C11 R04	CCCCGCAGCAAATGAAAAACCACGCTGAGAGCCACCCC

T C11 R05	CCCCTTCATCAATCTTTTCAAATTGTTATCCGCTTGGGTAAC
T C11 R06	CCCCAAATCAATATATGTGATGGAAACAGTACATCCCC
T C11 R07	CCCCGGAATCATGCGGTTTGATACGAGCCGGAAGCACGAAAGGG
T C11 R08	CCCCTATCATTCCAAGAACTCGGCTGTCTTTCCTCCCC
T C11 R09	CCCCAACGTCAACATTAATGGGGGTGCCTAATGAGTGTGCGGGC
T C11 R10	CCCCTACATACATAAAGGTAGCAAACGTAGAAAACCCC
T C11 R11	CCCCCGTTTTCATTTCCAGATTGCGTTGCGCTCACCCC
B C01 R01	CCCCGCATTGACCCACCACCCCCC
B C01 R02	CCCCATAAGTATAGCCCGGGTCGAGAGGGTTGATCCCC
B C01 R03	CCCCATAATAATAGAGCCACTTTGGGAA
B C01 R04	CCCCCAACCTAAAACGAAACCACTACGAAGGCACCCCC
B C01 R05	CCCCACGAGAAACAGAGCCGCACCAT
B C01 R06	CCCCGGGGGTAATAGTAAAAGAAGTTTTGCCAGACCCC
B C01 R07	CCCCTATAATGCGGAACCAGCACCAATG
B C01 R08	CCCCTTTATTTCAACGCAATTTGCGGGAGAAGCCCCCC
B C01 R09	CCCCAAACGTTACGTTTGCCATCAGTAG
B C01 R10	CCCCCAGGCTGCGCAACTGGCGCCATTCGCCATTCCCC
B C01 R11	CCCCCTGCCCGCTCGGCATCGTCAGACTGTAGCGCCCC
B C02 R01	GTCAGACGCAGGCGGATAAGTGCCAATAGGTG
B C02 R02	TATCACCGGAATAGAAAGGAACAACGTTGAAA
B C02 R03	ATCTCCAAGGTAAAATACGTAATGGAGGCAAA
B C02 R04	AGAATACAGCTGCTCATTCAGTGAACGAGTAG
B C02 R05	TAAATTGGGAGAGGCTTTTGCAAAATGTTTAG
B C02 R06	ACTGGATATTTGCGGATGGCTTAGCAACATGT
B C02 R07	TTTAAATAATGACCCTGTAATACTGGATAAAA
B C02 R08	ATTTTTAGGGAAGATTGTATAAGCGTTAAAAT
B C02 R09	TCGCATTACCGGAAACCAGGCAAATTGGGAAG
B C02 R10	GGCGATCGGAGCTAACTCACATTATCGGGAAA
B C02 R11	CCTGTCGTGCCAGCTGAAATGAAAACGATTTTTTGTTTCCCC
B C03 R01	AGGTTTAGGGGTTTTGCTCAGTACATTGGCCTGAGCAAGACCCC
B C03 R02	CTCATCTTAAGTTTCCATTAAACGAAAAAAGGCCCACAAGGTTGCTAT
B C03 R03	TACTGCGGTAAAAACCAAAATAGCGCTTGAGAGGGTAATCTGAATC
B C03 R04	TATATTTTTTGTACCAAAAACATTTGCAACTAAGAATTAACCAGAGCC
B C03 R05	CTCTTCGCGGCACCGCTTCTGGTGAATTTTTGAACATAAACAGCCATA
B C04 R01	ATAAATCGGATTAGGATTAGCGTACCGCCA
B C04 R02	CCCTCAGTAAACAACTTTCAACGGAGCCTT
B C04 R03	TAATTGTGACTTTTTCATGAGGTGACCCCC
B C04 R04	AGCGATTGGATATTCATTACCCTTTCAACT

B C04 R05	TTAATCATTTACCAGACGACGAAATCGTCA
B C04 R06	TAAATATCTTTAATTGCTCCTTTGTCTGGA
B C04 R07	AGTTTCAATAAAGCTAAATCGGAAATGCAA
B C04 R08	TGCCTGACCGGTTGATAATCAGGCTCATTT
B C04 R09	TTTAACCTCCAGCCAGCTTTCCTATTACGC
B C04 R10	CAGCTGGTAAAGTGTAAAGCCTAATCGGCC
B C04 R11	AACGCGCGGGGAGAGAATTACTATAAGAATAAACACCCCCC
B C05 R01	CCCTCAGAAGACTCCTCAAGAGAACTCATTAAAATATAAACCCC
B C05 R02	CGCGAAACGGCTTTGAGGACTAAAATCGGTTTGGCAGAGGTGATGCAA
B C05 R03	TCCCCCTCACTATCATAACCCTCGTTGTGAATGCCATATGAGAAAA
B C05 R04	TAGGTAAAAATAAAGCCTCAGAGCTTCCATATCAACGCTCTTTCATCT
B C05 R05	GGATGTGCCTCAGGAAGATCGCACAATAGGAATGCGTTATAAATACCG
B C06 R01	GCAGTCTCAAGTATTAAGAGGCTGACCGCCAC
B C06 R02	CCTCAGAGTTAGTAAATGAATTTTGCTTTCGA
B C06 R03	GGTGAATTTAGCAACGGCTACAGAAAAGTACA
B C06 R04	ACGGAGATGACCTTCATCAAGAGTGCGATTTT
B C06 R05	AAGAACTGGCATAGTAAGAGCAACAAATGCTT
B C06 R06	TAAACAGTGCAAACTCCAACAGGTATTCCCAA
B C06 R07	TTCTGCGAATTAGCAAAATTAAGCGATTCAAA
B C06 R08	AGGGTGAGATCGTAAAACTAGCATAAAATAAT
B C06 R09	TCGCGTCTCGACGACAGTATCGGCTGCAAGGC
B C06 R10	GATTAAGTCACAATTCCACACAACCGTATTGG
B C06 R11	GCGCCAGGGTGGTTTTTATAATCATCAGATGATGGCAACCCC
B C07 R01	CTCATTTTATTCTGAAACATGATGAATTTAGAGAAACACCCC
B C07 R02	TCGCCTGGCATCGGAACGAGGGTCTTAAACGAATATACCAATTCGA
B C07 R03	CGAGAATAAAAGGAATTACGAGGCTCATTAATTGCGTGAACGTT
B C07 R04	GAGACAGCAGGCAAGGCAAAGAACGAGTAGAAATTATTATTATCAT
B C07 R05	GCCAGGGGCCAGTTTGAGGGGAGGCCTTCCAATAATGGGAAGGAGC
B C08 R01	TCATACATCTATTTCGGAACCTATTCAGGGAT
B C08 R02	AGCAAGCCACGATCTAAAGTTTTGTACCGATA
B C08 R03	GTTGCGCCTCAGCAGCGAAAGACAATAAATTG
B C08 R04	TGTCGAAAACGGTGTACAGACCAGCAGGACGT
B C08 R05	TGGGAAGAGCAGATACATAACGCCGACCATAA
B C08 R06	ATCAAAAAAATTCGAGCTTCAAAGTTGACCAT
B C08 R07	TAGATACACATCCAATAAATCATATCAAATCA
B C08 R08	CCATCAATCTGGAGCAAACAAGAGAGCTTTCA
B C08 R09	TCAACATTTCGTAACCGTGCATCTTTTTCCCA
B C08 R10	GTCACGACAGCTGTTTCCTGTGTGACCAGTGA

B C08 R11	GACGGGCAACAGCTGACCGCCAGTGGTAATATCCAGAACCCC
B C09 R01	ACCCATGTAGTTAATGCCCCCTGCGGCTTTTGGAAAGCGTCCCC
B C09 R02	CTGCTCCATGCGGGATCGTCACCCGACAATGACTGGCCAATATAATCA
B C09 R03	TTACCCTGACCACATTCAACTAATAAAATCTACAGTCACTGTCCAT
B C09 R04	CAACCGTTATAGTAGTAGCATTAATTTCGCAACTGAAATGTACTTCTT
B C09 R05	ACGACGGCGGTGTAGATGGGCGCAAAATGTGATGGAAATACTGAGTAG
B C10 R01	CTGGTAAAGTGCCCGTATAAACACCGTAAC
B C10 R02	ACTGAGTCATTCCACAGACAGCATCGCCCA
B C10 R03	CGCATAAGTTAAAGGCCGCTTTTGTTACTT
B C10 R04	AGCCGGAGAACTGACCAACTTTAAACGAAC
B C10 R05	TAACGGATTGAGATTTAGGAATACTATTAT
B C10 R06	AGTCAGAGAAGCCCGAAAGACTTAACCTGT
B C10 R07	TTAGCTAGCATCAATTCTACTACTAGCTGA
B C10 R08	TAAATTATACAAAGGCTATCAGCAACCCGT
B C10 R09	CGGATTCGGGATAGGTCACGTTCAGTGCCA
B C10 R10	AGCTTGCGTACCGAGCTCGAATCACCGCCT
B C10 R11	GGCCCTGAGAGAGTTCAGCAGGCGAAAATC
B C11 R01	CCCCCAGTGCCTTGAGTAACTAAGTTTTGTCTATCACCCC
B C11 R02	CCCCACTACAACGCCTGTAGTTCGTCACCAGTACAACCCC
B C11 R03	CCCCCTGAGGCTTGCAGGGACCGATATACGTGGACTCCAACGTC
B C11 R04	CCCCCAATCATAAGGGAACCACGAGGCGCAGACGGTCCCC
B C11 R05	CCCCTAGAAAGATTCATCAGACAACATTCAGTTTGGAACAAGA
B C11 R06	CCCCTCAAAAAGATTAAGAGAGCAAAGCGGATTGCACCCC
B C11 R07	CCCCGCGAGCTGAAAAGGTGTATTTTCAAGAATAGCCCGAGATA
B C11 R08	CCCCCTATTTTTGAGAGATCATGCCGGAGAGGGTAGCCCC
B C11 R09	CCCCGCGGATTGACCGTAATTCCGTGGGCCGAAATCGGCAAAAT
B C11 R10	CCCCTCTAGAGGATCCCCGGATGCCTGCAGGTCGACCCCC

Table C.1: Sequences for double-layer origami.

leg	CATAACTTACTATCTATCTC
foot (3nt)	CTC
foot (5nt)	ATCTC
bridge (3nt)	CCT
bridge (4nt)	CCTA
bridge (5nt)	CCTAC
dock	TTTGTATTGT
guide	ACAATACAAA
S	CCAATCCTTTACACC

T	CTCCTC

Table C.2: Additional sequences used in Chapter 2.

Bri_T1R02C5	GAT ACA TTT CGC TTT TTT GAC CCT GTA AT
Bri_T1R02C5	GAT ACA TTT CGC TTT TTT GAC CCT GTA AT
Bri_T1R05C4	AAG CGA ACA ATT GCT GAA TAT AAT GCT GTA TTT TTT TGT GAG AAA GGC CGG
Bri_T1R05C4	AAG CGA ACA ATT GCT GAA TAT AAT GCT GTA TTT TTT TGT GAG AAA GGC CGG
Bri_T1R07C2	TGG ATA GCA AGC CCG ATT TTT AAT CGT AAA CGC CAT
Bri_T1R07C2	TGG ATA GCA AGC CCG ATT TTT AAT CGT AAA CGC CAT
Bri_T1R08C1	CAG AGG GGG TTT TGC CTT CCT GTA GCC AGC T
Bri_T1R08C1	CAG AGG GGG TTT TGC CTT CCT GTA GCC AGC T
Bri_T1R12C1	AGG ACA GAT GAT TTT TTC ACC AGT AGC ACC ATT ACC GAC TTG A
Bri_T1R12C1	AGG ACA GAT GAT TTT TTC ACC AGT AGC ACC ATT ACC GAC TTG A
Bri_T1R14C2	TGC CAC TAC TTT TTT TGC CAC CCT C
Bri_T1R14C2	TGC CAC TAC TTT TTT TGC CAC CCT C
Bri_T1R16C3	ACA ACC ATT TTT TCA TAC ATG GCT TTT AAG CGC A
Bri_T1R16C3	ACA ACC ATT TTT TCA TAC ATG GCT TTT AAG CGC A
Bri_T1R18C5	GAG AAT AGA AAG GAA CAA CTA TTT TCT CAA GAG AAG GA
Bri_T1R18C5	GAG AAT AGA AAG GAA CAA CTA TTT TCT CAA GAG AAG GA
Bri_T1R19C5	TGT CGT CTC AGC CCT CAT ATT TTT TTC GCC ACC CTC AGG TGT ATC
Bri_T1R19C5	TGT CGT CTC AGC CCT CAT ATT TTT TTC GCC ACC CTC AGG TGT ATC
Bri_T2R02C5	ACC GTA CTC AGG TTT TTG ATC TAA AGT TT
Bri_T2R02C5	ACC GTA CTC AGG TTT TTG ATC TAA AGT TT
Bri_T2R05C4	AGG AGT GTA AAC ATG AAA GTA TTA AGA GGC TTT TTT TGC GAA TAA TAA TTT
Bri_T2R05C4	AGG AGT GTA AAC ATG AAA GTA TTA AGA GGC TTT TTT TGC GAA TAA TAA TTT
Bri_T2R07C2	AGA ACC GCA TTT ACC GTT TTA CCG ATA TAT ACG TAA
Bri_T2R07C2	AGA ACC GCA TTT ACC GTT TTA CCG ATA TAT ACG TAA
Bri_T2R08C1	GAA CCG CCT CTT TAC CTA AAA CGA AAG AGG C
Bri_T2R08C1	GAA CCG CCT CTT TAC CTA AAA CGA AAG AGG C

Bri_T2R10C0	GGA ATT AGA GCT TTT TTT TCA GAC CAG GCG CGT TGG GAA GAT TTT TTT TCC AGG CAA AGC
Bri_T2R10C0	GGA ATT AGA GCT TTT TTT TCA GAC CAG GCG CGT TGG GAA GAT TTT TTT TCC AGG CAA AGC
Bri_T2R12C1	ATT AAG ACT CCT TTT TAA TAT ACA GTA ACA GTA CCG AAA TTG C
Bri_T2R12C1	ATT AAG ACT CCT TTT TAA TAT ACA GTA ACA GTA CCG AAA TTG C
Bri_T2R14C2	AAC TGA ACA TTT TTT TTG AAT AAC C
Bri_T2R14C2	AAC TGA ACA TTT TTT TTG AAT AAC C
Bri_T2R16C3	TTT TAT CTT TTT TAT CCA ATC GCA AGA GTT GGG T
Bri_T2R16C3	TTT TAT CTT TTT TAT CCA ATC GCA AGA GTT GGG T
Bri_T2R18C5	TTT TAT TTT CAT CGT AGG AAT TTT TAG CCT GTT TAG TA
Bri_T2R18C5	TTT TAT TTT CAT CGT AGG AAT TTT TAG CCT GTT TAG TA
Bri_T2R19C5	TAA TCG GCC ATC CTA ATT TTT TTT TTT TTT CGA GCC AAC AAC GCC
Bri_T2R19C5	TAA TCG GCC ATC CTA ATT TTT TTT TTT TTT CGA GCC AAC AAC GCC
Bri_T3R02C5	AAC ATG TAA TTT TTT TTG AAA CCA ATC AA
Bri_T3R02C5	AAC ATG TAA TTT TTT TTG AAA CCA ATC AA
Bri_T3R05C4	GCG AGA AAA TAA ACA CCG GAA TCA TAA TTA TTT TTT TCG CCC AAT AGC AAG
Bri_T3R05C4	GCG AGA AAA TAA ACA CCG GAA TCA TAA TTA TTT TTT TCG CCC AAT AGC AAG
Bri_T3R07C2	TTG CTT CTT ATA TGT ATT TTA CGC TAA CGG AGA ATT
Bri_T3R07C2	TTG CTT CTT ATA TGT ATT TTA CGC TAA CGG AGA ATT
Bri_T3R08C1	CAT AAA TCA ATT TAG TCA GAG GGT AAT TGA G
Bri_T3R08C1	CAT AAA TCA ATT TAG TCA GAG GGT AAT TGA G
Bri_T3R12C1	GAC AAC TCG TAT TTT TTC CTG TGT GAA ATT GTT ATC CGA GCT C
Bri_T3R12C1	GAC AAC TCG TAT TTT TTC CTG TGT GAA ATT GTT ATC CGA GCT C
Bri_T3R14C2	GCC ACG CTG TTT TTT TAC CAG TGA G
Bri_T3R14C2	GCC ACG CTG TTT TTT TAC CAG TGA G
Bri_T3R16C3	GCC AAC ATT TTT TCC ACT ATT AAA GAA ATA GGG T
Bri_T3R16C3	GCC AAC ATT TTT TCC ACT ATT AAA GAA ATA GGG T
Bri_T3R18C5	CAA ACT ATC GGC CTT GCT GGT TTT TGA GCT TGA CGG GG
Bri_T3R18C5	CAA ACT ATC GGC CTT GCT GGT TTT TGA GCT TGA CGG GG

Bri_T3R19C5	CTG TCC ATT TTT ATA ATC ATT TTT TTC TTA ATG CGC CCA CGC TGC
Bri_T3R19C5	CTG TCC ATT TTT ATA ATC ATT TTT TTC TTA ATG CGC CCA CGC TGC
Bri_T4R02C5	GCG TAA CCA CCA TTT TTG AGT AAA AGA GT
Bri_T4R02C5	GCG TAA CCA CCA TTT TTG AGT AAA AGA GT
Bri_T4R05C4	CCA ACG TCA TCG GAA CCC TAA AGG GAG CCC TTT TTT TGA ACA ATA TTA CCG
Bri_T4R05C4	CCA ACG TCA TCG GAA CCC TAA AGG GAG CCC TTT TTT TGA ACA ATA TTA CCG
Bri_T4R07C2	ACG GGC AAG TTC CAG TTT TTT CTG ACC TGC AAC AGT
Bri_T4R07C2	ACG GGC AAG TTC CAG TTT TTT CTG ACC TGC AAC AGT
Bri_T4R08C1	CCA GGG TGG TTT TGC AAA TGA AAA ATC TAA A
Bri_T4R08C1	CCA GGG TGG TTT TGC AAA TGA AAA ATC TAA A
Bri_T4R10C0	AAT CAT GGT CAT TTT TTT TTT TGC CCG AAC TCA GGT TTA ACT TTT TTT TCA GTA TGT TAG
Bri_T4R10C0	AAT CAT GGT CAT TTT TTT TTT TGC CCG AAC TCA GGT TTA ACT TTT TTT TCA GTA TGT TAG
Bri_T4R12C1	CCG CTT CTG GTT TTT TCG TTA ATA AAA CGA ACT AAA TTA TAC C
Bri_T4R12C1	CCG CTT CTG GTT TTT TCG TTA ATA AAA CGA ACT AAA TTA TAC C
Bri_T4R14C2	CAA AAA TAA TTT TTT TTG TTT AGA C
Bri_T4R14C2	CAA AAA TAA TTT TTT TTG TTT AGA C
Bri_T4R16C3	ACA AGA GTT TTT TTC GCG TTT TAA TTC AAA AAG A
Bri_T4R16C3	ACA AGA GTT TTT TTC GCG TTT TAA TTC AAA AAG A
Bri_T4R18C5	GAG TAA TGT GTA GGT AAA GAT TTT TTG TTT TAA ATA TG
Bri_T4R18C5	GAG TAA TGT GTA GGT AAA GAT TTT TTG TTT TAA ATA TG
Bri_T4R19C5	ACT TTT GCA TCG GTT GTA CTT TTT TTA ACC TGT TTA GGA CCA TTA
Bri_T4R19C5	ACT TTT GCA TCG GTT GTA CTT TTT TTA ACC TGT TTA GGA CCA TTA
Reg_T1R01C6	TCA TTT GCT AAT AGT AGT AGC ATT
Reg_T1R01C6	TCA TTT GCT AAT AGT AGT AGC ATT
Reg_T1R03C5	CAA CTA AAG TAC GGT GGG ATG GCT
Reg_T1R03C5	CAA CTA AAG TAC GGT GGG ATG GCT
Reg_T1R03C6	TTT CAT TGA GTA GAT TTA GTT TCT ATA TTT
Reg_T1R03C6	TTT CAT TGA GTA GAT TTA GTT TCT ATA TTT

Reg_T1R04C5	TAG AGC TTC AGA CCG GAA GCA AAC CTA TTA TA
Reg_T1R04C5	TAG AGC TTC AGA CCG GAA GCA AAC CTA TTA TA
Reg_T1R05C6	GTC AGG AAG AGG TCA TTT TTG CTC TGG AAG
Reg_T1R05C6	GTC AGG AAG AGG TCA TTT TTG CTC TGG AAG
Reg_T1R06C3	TTA AGA GGG TCC AAT ACT GCG GAT AGC GAG
Reg_T1R06C3	TTA AGA GGG TCC AAT ACT GCG GAT AGC GAG
Reg_T1R06C5	GTC AGA AGA TTG AAT CCC CCT CAA CCT CGT TT
Reg_T1R06C5	GTC AGA AGA TTG AAT CCC CCT CAA CCT CGT TT
Reg_T1R07C4	AAA TAT TCC AAA GCG GAT TGC ATC GAG CTT CA
Reg_T1R07C4	AAA TAT TCC AAA GCG GAT TGC ATC GAG CTT CA
Reg_T1R07C6	AAC AGT TAG GTC TTT ACC CTG ATC CAA CAG
Reg_T1R07C6	AAC AGT TAG GTC TTT ACC CTG ATC CAA CAG
Reg_T1R08C3	AGG CTT TTC AGG TAG AAA GAT TCA ATT ACC
Reg_T1R08C3	AGG CTT TTC AGG TAG AAA GAT TCA ATT ACC
Reg_T1R08C5	ACC AGA CGG AAT ACC ACA TTC AAC GAG ATG GT
Reg_T1R08C5	ACC AGA CGG AAT ACC ACA TTC AAC GAG ATG GT
Reg_T1R09C2	CAT TAT TAG CAA AAG AAG TTT TGC
Reg_T1R09C2	CAT TAT TAG CAA AAG AAG TTT TGC
Reg_T1R09C4	AGA TTT AGA CGA TAA AAA CCA AAA ATC GTC AT
Reg_T1R09C4	AGA TTT AGA CGA TAA AAA CCA AAA ATC GTC AT
Reg_T1R09C6	ATA CAT ACA ACA CTA TCA TAA CAT GCT TTA
Reg_T1R09C6	ATA CAT ACA ACA CTA TCA TAA CAT GCT TTA
Reg_T1R10C1	AGT CAG GAC ATA GGC TGG CTG ACC TTT GAA AG
Reg_T1R10C1	AGT CAG GAC ATA GGC TGG CTG ACC TTT GAA AG
Reg_T1R10C3	TTA TGC GAT TGA CAA GAA CCG GAG GTC AAT
Reg_T1R10C3	TTA TGC GAT TGA CAA GAA CCG GAG GTC AAT
Reg_T1R10C5	TTA ATT TCC AAC GTA ACA AAG CTG TCC ATG TT
Reg_T1R10C5	TTA ATT TCC AAC GTA ACA AAG CTG TCC ATG TT
Reg_T1R11C2	GAG TAA TCT TTT AAG AAC TGG CTC CGG AAC AA
Reg_T1R11C2	GAG TAA TCT TTT AAG AAC TGG CTC CGG AAC AA
Reg_T1R11C4	ACC CAA ATA ACT TTA ATC ATT GTG ATC AGT TG
Reg_T1R11C4	ACC CAA ATA ACT TTA ATC ATT GTG ATC AGT TG
Reg_T1R11C6	GTG AAT ATA GTA AAT TGG GCT TTA ATG CAG
Reg_T1R11C6	GTG AAT ATA GTA AAT TGG GCT TTA ATG CAG
Reg_T1R12C3	CAT AAG GGA CAC TAA AAC ACT CAC ATT AAA
Reg_T1R12C3	CAT AAG GGA CAC TAA AAC ACT CAC ATT AAA
Reg_T1R12C5	ACT TAG CCA TTA TAC CAA GCG CGA GAG GAC TA
Reg_T1R12C5	ACT TAG CCA TTA TAC CAA GCG CGA GAG GAC TA

Reg_T1R13C2	AAA AGA ATA ACC GAA CTG ACC AAC TTC ATC AA
Reg_T1R13C2	AAA AGA ATA ACC GAA CTG ACC AAC TTC ATC AA
Reg_T1R13C4	CCC CAG CGG GAA CGA GGC GCA GAC TAT TCA TT
Reg_T1R13C4	CCC CAG CGG GAA CGA GGC GCA GAC TAT TCA TT
Reg_T1R13C6	ACA ACG GAA ATC CGC GAC CTG CCT CAT TCA
Reg_T1R13C6	ACA ACG GAA ATC CGC GAC CTG CCT CAT TCA
Reg_T1R14C3	CGG GTA AAA TTC GGT CGC TGA GGA ATG ACA
Reg_T1R14C3	CGG GTA AAA TTC GGT CGC TGA GGA ATG ACA
Reg_T1R14C5	AAG ACT TTG GCC GCT TTT GCG GGA TTA AAC AG
Reg_T1R14C5	AAG ACT TTG GCC GCT TTT GCG GGA TTA AAC AG
Reg_T1R15C4	GAG TTA AAT TCA TGA GGA AGT TTC TCT TTG AC
Reg_T1R15C4	GAG TTA AAT TCA TGA GGA AGT TTC TCT TTG AC
Reg_T1R15C6	CTC AGC AGG CTA CAG AGG CTT TAA CAA AGT
Reg_T1R15C6	CTC AGC AGG CTA CAG AGG CTT TAA CAA AGT
Reg_T1R16C5	CTT GAT ACT GAA AAT CTC CAA AAA AGC GGA GT
Reg_T1R16C5	CTT GAT ACT GAA AAT CTC CAA AAA AGC GGA GT
Reg_T1R17C4	TTT CAC GTC GAT AGT TGC GCC GAC CTT GCA GG
Reg_T1R17C4	TTT CAC GTC GAT AGT TGC GCC GAC CTT GCA GG
Reg_T1R17C6	CAA AAG GTT CGA GGT GAA TTT CTC GTC ACC
Reg_T1R17C6	CAA AAG GTT CGA GGT GAA TTT CTC GTC ACC
Reg_T1R19C6	GTT AGT AAC TTT CAA CAG TTT CAA AGG CTC
Reg_T1R19C6	GTT AGT AAC TTT CAA CAG TTT CAA AGG CTC
Reg_T1R21C5	CCA TGT ACC GTA ACA CTG TAG CAT TCC ACA GAT TCC AGA C
Reg_T1R21C5	CCA TGT ACC GTA ACA CTG TAG CAT TCC ACA GAT TCC AGA C
Reg_T2R01C6	ACC CTC ATT CAG GGA TAG CAA GCC
Reg_T2R01C6	ACC CTC ATT CAG GGA TAG CAA GCC
Reg_T2R03C5	TTA GGA TTA GCG GGG TGG AAC CTA
Reg_T2R03C5	TTA GGA TTA GCG GGG TGG AAC CTA
Reg_T2R03C6	GTA CCA GGT ATA GCC CGG AAT AGA ACC GCC
Reg_T2R03C6	GTA CCA GGT ATA GCC CGG AAT AGA ACC GCC
Reg_T2R04C5	TTA TTC TGA CTG GTA ATA AGT TTT AAC AAA TA
Reg_T2R04C5	TTA TTC TGA CTG GTA ATA AGT TTT AAC AAA TA
Reg_T2R05C6	CAG TGC CCC CCC TGC CTA TTT CTT TGC TCA
Reg_T2R05C6	CAG TGC CCC CCC TGC CTA TTT CTT TGC TCA
Reg_T2R06C3	GTC TCT GAC ACC CTC AGA GCC ACA TCA AAA
Reg_T2R06C3	GTC TCT GAC ACC CTC AGA GCC ACA TCA AAA

Reg_T2R06C5	AAT CCT CAA CCA GAA CCA CCA CCA GCC CCC TT
Reg_T2R06C5	AAT CCT CAA CCA GAA CCA CCA CCA GCC CCC TT
Reg_T2R07C4	GAG CCG CCT TAA AGC CAG AAT GGA GAT GAT AC
Reg_T2R07C4	GAG CCG CCT TAA AGC CAG AAT GGA GAT GAT AC
Reg_T2R07C6	GCC AGC AGC CTT GAT ATT CAC AAA CGG GGT
Reg_T2R07C6	GCC AGC AGC CTT GAT ATT CAC AAA CGG GGT
Reg_T2R08C3	TCA CCG GAA ACG TCA CCA ATG AAT TAT TCA
Reg_T2R08C3	TCA CCG GAA ACG TCA CCA ATG AAT TAT TCA
Reg_T2R08C5	ATT AGC GTC CGT AAT CAG TAG CGA ATT GAG GG
Reg_T2R08C5	ATT AGC GTC CGT AAT CAG TAG CGA ATT GAG GG
Reg_T2R09C2	AGG CCG GAA CCA GAG CCA CCA CCG
Reg_T2R09C2	AGG CCG GAA CCA GAG CCA CCA CCG
Reg_T2R09C4	TAG CAG CAT TGC CAT CTT TTC ATA CAC CCT CA
Reg_T2R09C4	TAG CAG CAT TGC CAT CTT TTC ATA CAC CCT CA
Reg_T2R09C6	AGT TTG CGC ATT TTC GGT CAT AGA GCC GCC
Reg_T2R09C6	AGT TTG CGC ATT TTC GGT CAT AGA GCC GCC
Reg_T2R10C1	GCC ATT TGC AAA CGT AGA AAA TAC CTG GCA TG
Reg_T2R10C1	GCC ATT TGC AAA CGT AGA AAA TAC CTG GCA TG
Reg_T2R10C3	TTA AAG GTA CAT ATA AAA GAA ACA AAC GCA
Reg_T2R10C3	TTA AAG GTA CAT ATA AAA GAA ACA AAC GCA
Reg_T2R10C5	AGG GAA GGA TAA GTT TAT TTT GTC AGC CGA AC
Reg_T2R10C5	AGG GAA GGA TAA GTT TAT TTT GTC AGC CGA AC
Reg_T2R11C2	AGG TGG CAG AAT TAT CAC CGT CAC CAT TAG CA
Reg_T2R11C2	AGG TGG CAG AAT TAT CAC CGT CAC CAT TAG CA
Reg_T2R11C4	ACC ACG GAT AAA TAT TGA CGG AAA ACC ATC GA
Reg_T2R11C4	ACC ACG GAT AAA TAT TGA CGG AAA ACC ATC GA
Reg_T2R11C6	TAG AAA AGG CGA CAT TCA ACC GCA GAA TCA
Reg_T2R11C6	TAG AAA AGG CGA CAT TCA ACC GCA GAA TCA
Reg_T2R12C3	ATA ATA ACT CAG AGA GAT AAC CCG AAG CGC
Reg_T2R12C3	ATA ATA ACT CAG AGA GAT AAC CCG AAG CGC
Reg_T2R12C5	AAA GTT ACG CCC AAT AAT AAG AGC AGC CTT TA
Reg_T2R12C5	AAA GTT ACG CCC AAT AAT AAG AGC AGC CTT TA
Reg_T2R13C2	CGC TAA TAG GAA TAC CCA AAA GAA ATA CAT AA
Reg_T2R13C2	CGC TAA TAG GAA TAC CCA AAA GAA ATA CAT AA
Reg_T2R13C4	TGA GTT AAC AGA AGG AAA CCG AGG GCA AAG AC
Reg_T2R13C4	TGA GTT AAC AGA AGG AAA CCG AGG GCA AAG AC
Reg_T2R13C6	ATG AAA TGA AAA GTA AGC AGA TAC AAT CAA
Reg_T2R13C6	ATG AAA TGA AAA GTA AGC AGA TAC AAT CAA

Reg_T2R14C3	ATT AGA CGG AGC GTC TTT CCA GAG CTA CAA
Reg_T2R14C3	ATT AGA CGG AGC GTC TTT CCA GAG CTA CAA
Reg_T2R14C5	CAG AGA GAA CAA AAT AAA CAG CCA TTA AAT CA
Reg_T2R14C5	CAG AGA GAA CAA AAT AAA CAG CCA TTA AAT CA
Reg_T2R15C4	TGC CAG TTA TAA CAT AAA AAC AGG ACA AGA AT
Reg_T2R15C4	TGC CAG TTA TAA CAT AAA AAC AGG ACA AGA AT
Reg_T2R15C6	ATC CCA AAA AAA TGA AAA TAG CAA GAA ACA
Reg_T2R15C6	ATC CCA AAA AAA TGA AAA TAG CAA GAA ACA
Reg_T2R16C5	AGA TTA GTA TAT AGA AGG CTT ATC CAA GCC GT
Reg_T2R16C5	AGA TTA GTA TAT AGA AGG CTT ATC CAA GCC GT
Reg_T2R17C4	CAA ATC AGT GCT ATT TTG CAC CCA GCC TAA TT
Reg_T2R17C4	CAA ATC AGT GCT ATT TTG CAC CCA GCC TAA TT
Reg_T2R17C6	TAA GAA CGG AGG TTT TGA AGC CTA TTA TTT
Reg_T2R17C6	TAA GAA CGG AGG TTT TGA AGC CTA TTA TTT
Reg_T2R19C6	CTT ATC ACT CAT CGA GAA CAA GCG GTA TTC
Reg_T2R19C6	CTT ATC ACT CAT CGA GAA CAA GCG GTA TTC
Reg_T2R21C5	AGC TAA TGC AGA ACG CGA GAA AAA TAA TAT CCT GTC TTT C
Reg_T2R21C5	AGC TAA TGC AGA ACG CGA GAA AAA TAA TAT CCT GTC TTT C
Reg_T3R01C6	AGA ATA TCA GAC GAC GAC AAT AAA
Reg_T3R01C6	AGA ATA TCA GAC GAC GAC AAT AAA
Reg_T3R03C5	TCA TAT GCG TTA TAC AAA GGC GTT
Reg_T3R03C5	TCA TAT GCG TTA TAC AAA GGC GTT
Reg_T3R03C6	CCA GTA TGA ATC GCC ATA TTT AGT AAT AAG
Reg_T3R03C6	CCA GTA TGA ATC GCC ATA TTT AGT AAT AAG
Reg_T3R04C5	AAA TAA GAA CTT TTT CAA ATA TAT CTG AGA GA
Reg_T3R04C5	AAA TAA GAA CTT TTT CAA ATA TAT CTG AGA GA
Reg_T3R05C6	ATT TCA TGA CCG TGT GAT AAA TAA TTC TTA
Reg_T3R05C6	ATT TCA TGA CCG TGT GAT AAA TAA TTC TTA
Reg_T3R06C3	TAT ATA ACG TAA ATC GTC GCT ATA TTT GAA
Reg_T3R06C3	TAT ATA ACG TAA ATC GTC GCT ATA TTT GAA
Reg_T3R06C5	CTA CCT TTA GAA TCC TTG AAA ACA AGA AAA CA
Reg_T3R06C5	CTA CCT TTA GAA TCC TTG AAA ACA AGA AAA CA
Reg_T3R07C4	TTT CCC TTT TAA CCT CCG GCT TAG CAA AGA AC
Reg_T3R07C4	TTT CCC TTT TAA CCT CCG GCT TAG CAA AGA AC
Reg_T3R07C6	GCT TAG AAT CAA AAT CAT AGG TTT TAG TTA
Reg_T3R07C6	GCT TAG AAT CAA AAT CAT AGG TTT TAG TTA

Reg_T3R08C3	TTA CCT TTA CAA TAA CGG ATT CGC AAA ATT
Reg_T3R08C3	TTA CCT TTA CAA TAA CGG ATT CGC AAA ATT
Reg_T3R08C5	AAA TTA ATA CCA AGT TAC AAA ATC CTG AAT AA
Reg_T3R08C5	AAA TTA ATA CCA AGT TAC AAA ATC CTG AAT AA
Reg_T3R09C2	CGG GAG AAT TTA ATG GAA ACA GTA
Reg_T3R09C2	CGG GAG AAT TTA ATG GAA ACA GTA
Reg_T3R09C4	CTT TGA ATT ACA TTT AAC AAT TTC TAA TTA AT
Reg_T3R09C4	CTT TGA ATT ACA TTT AAC AAT TTC TAA TTA AT
Reg_T3R09C6	GCG AAT TAT GAA ACA AAC ATC ATA GCG ATA
Reg_T3R09C6	GCG AAT TAT GAA ACA AAC ATC ATA GCG ATA
Reg_T3R10C1	GTA GAT TTG TTA TTA ATT TTA AAA AAC AAT TC
Reg_T3R10C1	GTA GAT TTG TTA TTA ATT TTA AAA AAC AAT TC
Reg_T3R10C3	ATT TGC ACC ATT TTG CGG AAC AAA TTT GAG
Reg_T3R10C3	ATT TGC ACC ATT TTG CGG AAC AAA TTT GAG
Reg_T3R10C5	TGG AAG GGA GCG GAA TTA TCA TCA ACT AAT AG
Reg_T3R10C5	TGG AAG GGA GCG GAA TTA TCA TCA ACT AAT AG
Reg_T3R11C2	AAC ATT ATG TAA AAC AGA AAT AAA TTT TAC AT
Reg_T3R11C2	AAC ATT ATG TAA AAC AGA AAT AAA TTT TAC AT
Reg_T3R11C4	CCA GAA GGT TAG AAC CTA CCA TAT CCT GAT TG
Reg_T3R11C4	CCA GAA GGT TAG AAC CTA CCA TAT CCT GAT TG
Reg_T3R11C6	ATT ATC AGT TTG GAT TAT ACT TGC GCA GAG
Reg_T3R11C6	ATT ATC AGT TTG GAT TAT ACT TGC GCA GAG
Reg_T3R12C3	GAT TTA GAT TGC TGA ACC TCA AAG TAT TAA
Reg_T3R12C3	GAT TTA GAT TGC TGA ACC TCA AAG TAT TAA
Reg_T3R12C5	ATT AGA GCA ATA TCT GGT CAG TTG CAG CAG AA
Reg_T3R12C5	ATT AGA GCA ATA TCT GGT CAG TTG CAG CAG AA
Reg_T3R13C2	GCA TCA CCA GTA TTA GAC TTT ACA GTT TGA GT
Reg_T3R13C2	GCA TCA CCA GTA TTA GAC TTT ACA GTT TGA GT
Reg_T3R13C4	CCT CAA TCC GTC AAT AGA TAA TAC AGA AAC CA
Reg_T3R13C4	CCT CAA TCC GTC AAT AGA TAA TAC AGA AAC CA
Reg_T3R13C6	ACA GTT GTT AGG AGC ACT AAC ATA TTC CTG
Reg_T3R13C6	ACA GTT GTT AGG AGC ACT AAC ATA TTC CTG
Reg_T3R14C3	CAC CGC CTG AAA GCG TAA GAA TAC ATT CTG
Reg_T3R14C3	CAC CGC CTG AAA GCG TAA GAA TAC ATT CTG
Reg_T3R14C5	GAT AAA ACT TTT TGA ATG GCT ATT TTC ACC AG
Reg_T3R14C5	GAT AAA ACT TTT TGA ATG GCT ATT TTC ACC AG
Reg_T3R15C4	AGA CAA TAA GAG GTG AGG CGG TCA TAT CAA AC
Reg_T3R15C4	AGA CAA TAA GAG GTG AGG CGG TCA TAT CAA AC

Reg_T3R15C6	ATG CGC GTA CCG AAC GAA CCA CGC AAA TCA
Reg_T3R15C6	ATG CGC GTA CCG AAC GAA CCA CGC AAA TCA
Reg_T3R16C5	TCA CAC GAT GCA ACA GGA AAA ACG GAA GAA CT
Reg_T3R16C5	TCA CAC GAT GCA ACA GGA AAA ACG GAA GAA CT
Reg_T3R17C4	CCA GCC ATC CAG TAA TAA AAG GGA CGT GGC AC
Reg_T3R17C4	CCA GCC ATC CAG TAA TAA AAG GGA CGT GGC AC
Reg_T3R17C6	AAT ACC TAT TTA CAT TGG CAG AAG TCT TTA
Reg_T3R17C6	AAT ACC TAT TTA CAT TGG CAG AAG TCT TTA
Reg_T3R19C6	TTA ACC GTC ACT TGC CTG AGT ACT CAT GGA
Reg_T3R19C6	TTA ACC GTC ACT TGC CTG AGT ACT CAT GGA
Reg_T3R21C5	CTA AAC AGG AGG CCG ATA ATC CTG AGA AGT GTC ACG CAA A
Reg_T3R21C5	CTA AAC AGG AGG CCG ATA ATC CTG AGA AGT GTC ACG CAA A
Reg_T4R01C6	GCG CGT ACT TTC CTC GTT AGA ATC
Reg_T4R01C6	GCG CGT ACT TTC CTC GTT AGA ATC
Reg_T4R03C5	AAA GCC GGC GAA CGT GTG CCG TAA
Reg_T4R03C5	AAA GCC GGC GAA CGT GTG CCG TAA
Reg_T4R03C6	GGA AGG GGG CAA GTG TAG CGG TGC TAC AGG
Reg_T4R03C6	GGA AGG GGG CAA GTG TAG CGG TGC TAC AGG
Reg_T4R04C5	AGC ACT AAA AAG GGC GAA AAA CCG AAA TCC CT
Reg_T4R04C5	AGC ACT AAA AAG GGC GAA AAA CCG AAA TCC CT
Reg_T4R05C6	GGC GAT GTT TTT GGG GTC GAG GGC GAG AAA
Reg_T4R05C6	GGC GAT GTT TTT GGG GTC GAG GGC GAG AAA
Reg_T4R06C3	TGA GTG TTC AGC TGA TTG CCC TTG CGC GGG
Reg_T4R06C3	TGA GTG TTC AGC TGA TTG CCC TTG CGC GGG
Reg_T4R06C5	TAT AAA TCG AGA GTT GCA GCA AGC GTC GTG CC
Reg_T4R06C5	TAT AAA TCG AGA GTT GCA GCA AGC GTC GTG CC
Reg_T4R07C4	GGC CCT GAA AAA GAA TAG CCC GAG CGT GGA CT
Reg_T4R07C4	GGC CCT GAA AAA GAA TAG CCC GAG CGT GGA CT
Reg_T4R07C6	CTG GTT TGT TCC GAA ATC GGC ATC TAT CAG
Reg_T4R07C6	CTG GTT TGT TCC GAA ATC GGC ATC TAT CAG
Reg_T4R08C3	GAG AGG CGA CAA CAT ACG AGC CGC TGC AGG
Reg_T4R08C3	GAG AGG CGA CAA CAT ACG AGC CGC TGC AGG
Reg_T4R08C5	AGC TGC ATA GCC TGG GGT GCC TAA GTA AAA CG
Reg_T4R08C5	AGC TGC ATA GCC TGG GGT GCC TAA GTA AAA CG
Reg_T4R09C2	AAT TCC ACG TTT GCG TAT TGG GCG
Reg_T4R09C2	AAT TCC ACG TTT GCG TAT TGG GCG

Reg_T4R09C4	AAG TGT AAT AAT GAA TCG GCC AAC CAC CGC CT
Reg_T4R09C4	AAG TGT AAT AAT GAA TCG GCC AAC CAC CGC CT
Reg_T4R09C6	CTA ACT CCC AGT CGG GAA ACC TGG TCC ACG
Reg_T4R09C6	CTA ACT CCC AGT CGG GAA ACC TGG TCC ACG
Reg_T4R10C1	GAA TTC GTG CCA TTC GCC ATT CAG TTC CGG CA
Reg_T4R10C1	GAA TTC GTG CCA TTC GCC ATT CAG TTC CGG CA
Reg_T4R10C3	TCG ACT CTG AAG GGC GAT CGG TGC GGC CTC
Reg_T4R10C3	TCG ACT CTG AAG GGC GAT CGG TGC GGC CTC
Reg_T4R10C5	ACG GCC AGT ACG CCA GCT GGC GAA CAT CTG CC
Reg_T4R10C5	ACG GCC AGT ACG CCA GCT GGC GAA CAT CTG CC
Reg_T4R11C2	ACT GTT GGA GAG GAT CCC CGG GTA CCG CTC AC
Reg_T4R11C2	ACT GTT GGA GAG GAT CCC CGG GTA CCG CTC AC
Reg_T4R11C4	TTC GCT ATT GCC AAG CTT GCA TGC GAA GCA TA
Reg_T4R11C4	TTC GCT ATT GCC AAG CTT GCA TGC GAA GCA TA
Reg_T4R11C6	GTG CTG CCC CAG TCA CGA CGT TTG AGT GAG
Reg_T4R11C6	GTG CTG CCC CAG TCA CGA CGT TTG AGT GAG
Reg_T4R12C3	AGG AAG ATC ATT AAA TGT GAG CGT TTT TAA
Reg_T4R12C3	AGG AAG ATC ATT AAA TGT GAG CGT TTT TAA
Reg_T4R12C5	AGT TTG AGA TTC TCC GTG GGA ACA ATT CGC AT
Reg_T4R12C5	AGT TTG AGA TTC TCC GTG GGA ACA ATT CGC AT
Reg_T4R13C2	TTC ATC AAC GCA CTC CAG CCA GCT GCT GCG CA
Reg_T4R13C2	TTC ATC AAC GCA CTC CAG CCA GCT GCT GCG CA
Reg_T4R13C4	CCC GTC GGG GGA CGA CGA CAG TAT CGG GCC TC
Reg_T4R13C4	CCC GTC GGG GGA CGA CGA CAG TAT CGG GCC TC
Reg_T4R13C6	ATT GAC CCG CAT CGT AAC CGT GAG GGG GAT
Reg_T4R13C6	ATT GAC CCG CAT CGT AAC CGT GAG GGG GAT
Reg_T4R14C3	CCA ATA GGA AAC TAG CAT GTC AAG GAG CAA
Reg_T4R14C3	CCA ATA GGA AAC TAG CAT GTC AAG GAG CAA
Reg_T4R14C5	TAA ATT TTT GAT AAT CAG AAA AGC ACA AAG GC
Reg_T4R14C5	TAA ATT TTT GAT AAT CAG AAA AGC ACA AAG GC
Reg_T4R15C4	ACC CCG GTT GTT AAA TCA GCT CAT AGT AAC AA
Reg_T4R15C4	ACC CCG GTT GTT AAA TCA GCT CAT AGT AAC AA
Reg_T4R15C6	CAG GAA GTA ATA TTT TGT TAA AAA CGG CGG
Reg_T4R15C6	CAG GAA GTA ATA TTT TGT TAA AAA CGG CGG
Reg_T4R16C5	TAT CAG GTA AAT CAC CAT CAA TAT CAA TGC CT
Reg_T4R16C5	TAT CAG GTA AAT CAC CAT CAA TAT CAA TGC CT
Reg_T4R17C4	AGA CAG TCC ATT GCC TGA GAG TCT TCA TAT GT
Reg_T4R17C4	AGA CAG TCC ATT GCC TGA GAG TCT TCA TAT GT

Reg_T4R17C6	ACC GTT CAT TTT TGA GAG ATC TCC CAA AAA
Reg_T4R17C6	ACC GTT CAT TTT TGA GAG ATC TCC CAA AAA
Reg_T4R19C6	CCT TTA TCA TAT ATT TTA AAT GGA TAT TCA
Reg_T4R19C6	CCT TTA TCA TAT ATT TTA AAT GGA TAT TCA
Reg_T4R21C5	AAT CAT ACA GGC AAG GCA GAG CAT AAA GCT AAG GGA GAA G
Reg_T4R21C5	AAT CAT ACA GGC AAG GCA GAG CAT AAA GCT AAG GGA GAA G

Table C.3: Sequences for single-layer origami.

INDEX