UNIVERSITY OF MINNESOTA

MASTER'S THESIS

Random Access DNA Registers

Author:
Jake KASLEWICZ

Supervisor:
Dr. Marc RIEDEL

A thesis submitted in fulfillment of the requirements
for the degree of Master's in Electrical and Computer Engineering

in the

The Circuits and Biology Lab
Electrical and Computer Engineering

May 28, 2024

UNIVERSITY OF MINNESOTA

Abstract

Marc Riedel

Electrical and Computer Engineering

Master's in Electrical and Computer Engineering

Random Access DNA Registers

by Jake KASLEWICZ

Recent advancements in DNA-based data storage have primarily focused on maximizing data density, often at the expense of operational flexibility. Traditional encoding schemes require complete resequencing for data modification, while current computational approaches rely on slow reaction primitives. This research addresses the need for an error-resistant system that enables complex computational tasks directly on DNA without compromising local random access memory operations. We propose a novel encoding strategy that stores data in individual register strands. Each register strand encodes both the value and address of the data in two functionally separate subsequences, providing additional parameters to control error accumulation during operations. Simulation results demonstrate that the system can selectively retrieve individual register strands, enabling the precise modification of single values. It can also perform logical AND operations to compute across adjacent stored values. These operations facilitate the transfer of stored data from one system to another. This capability suggests that a modular network of systems could potentially scale to achieve the highly parallel computation required to overcome the inherently slow operation speed of DNA computing.

Acknowledgements

I would like to thank my advisor, Marc Riedel, for introducing me to the complex world of DNA computation and storage. Thanks to Erik Winfree, I was able to visit Japan and meet many experts in the field. Thanks to Boyan Beronov for supporting the project from the beginning. Multistrand would still be collecting dust if it weren't for your help.

I would also like to thank Matt Oehler, Chris Stallard, Rishi Gulati, and Adam Justin for listening to my new ideas every single day. There are many others who have found themselves stuck in a conversation about DNA strands.

Caeden Windschitl for providing feedback on the report.

Finally, I would like to thank Georia Molldrem, for supporting me through the entire process.

Contents

List of Figures

Chapter 1

Introduction

One of the most pressing challenges in chip design today is the end of Moore's Law. Traditionally, Moore's Law predicts a doubling of chip densities every two years, a trend that has now stagnated due to the physical limitations of transistor scaling[11]. As a result, the focus has shifted towards enhancing parallel processing capabilities and advancing miniaturization techniques[11]. Amid these explorations, DNA-based technology is exceptionally promising. DNA can achieve unprecedented data densities—up to 215 petabytes per gram—while capitalizing on the inherent advantages of parallel processing[19]. This paradigm shift challenges conventional computing frameworks and opens exciting new possibilities for integrating biological mechanisms into future computing architectures.

The prevailing method of DNA storage converts data from a binary representation (0/1) to a base-4 representation (A/C/T/G). This base-4 representation is then synthesized into the nucleotides of DNA strands, which are suspended in solution[4, 17]. Next-generation sequencing (NGS) technologies are used to extract the encoded data from the DNA strands[16, 19]. However, repeatedly sequencing the encoded data for each step of computation is time-consuming. Additionally, modifying a single bit requires the replication of other stored values, increasing the risk of errors[19, 5].

In response to these limitations, there has been a shift towards encoding schemes that support computation directly on DNA. These schemes often sacrifice data density to exploit the medium's parallel capabilities[8, 13, 6]. Despite their potential, these models face significant scalability issues, primarily due to error accumulation. Furthermore, reading data from

these schemes typically requires multiple rounds of instructions, making it difficult to modify singular stored values[8, 15, 13].

The primary goal of this thesis is to develop a scalable DNA system that supports fast input/output operations essential for storage while maintaining computational capabilities. A key aspect of scalability is mitigating error-prone pathways to maintain accuracy. Additionally, the design framework should be flexible enough to operate under different temperatures and concentrations.

To achieve these goals, we propose a novel encoding scheme that utilizes a single DNA strand for each bit of data, referred to as a *reg* (register) strand. The *reg* strand is divided into subsections, known as domains. One domain encodes the value through a specific sequence of nucleotides, while the other domain encodes the storage location. These domains are further divided into subdomains, which introduce design parameters to control DNA reactions for modifying and comparing data across different *reg* strands.

The system is considered successful if individual *reg* strands can be isolated from the main solution. Additionally, the system must be able to compare stored values across different *reg* strands. Constraints must be identified for the design parameters to determine the concentrations and temperatures at which the system functions correctly.

In this thesis, Chapter 2 provides background information, reviews models from previous literature, and formulates the problem addressed. Chapter 3 outlines the design methodology developed to tackle the identified issues, detailing the theoretical framework and practical considerations. Chapter 4 describes the methods used to simulate the basic instruction set, including the computational tools and parameters employed. Chapter 5 presents the results of these simulations, analyzing their implications and evaluating the performance of the proposed system. Finally, Chapter 6 summarizes the conclusions drawn from this research and suggests directions for future work.

Chapter 2

Background & Problem Identification

2.1 Strands

DNA strands are chains of nucleotides, or DNA bases. Each strand is uniquely identified by the specific sequence of nucleotides that compose the larger DNA molecule. These sequences determine a strand's properties and its interactions with other strands in a system. The precise arrangement of adenine (A), thymine (T), cytosine (C), and guanine (G) dictates how a strand will bind with complementary sequences through Watson-Crick pairing, forming the basis for complex molecular interactions[17, 4].

2.2 Domains and Sequence Design

In DNA-based systems, subsequences of nucleotides within a strand are organized into functional units known as domains. There are two primary types of domains utilized: toehold and migration domains. Toeholds, typically comprising 6-10 bases, are essential for initiating complex molecular interactions in conjunction with migration domains[12, 22, 21]. The toehold domains significantly control the kinetics of the reaction; the speed of the reaction is closely tied to the length and composition of the toeholds. For instance, a higher GC content results in stronger toeholds that accelerate the reaction due to more stable hydrogen bonding interactions between complementary strands[22, 6, 12].

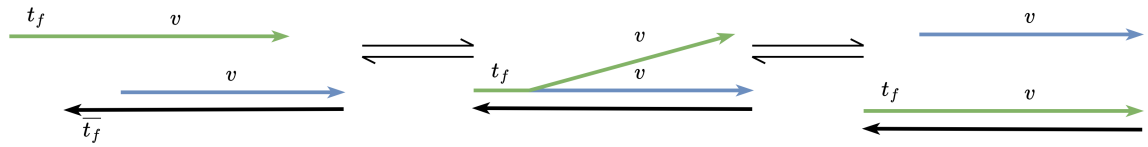


FIGURE 2.1: The reaction kinetics of the strand displacement are controlled primarily by the forward toehold, t_f. The migration domain is labeled with a v.

Similar to the binding specificity of individual nucleotides, domains are designed to bind only with their complement. To maintain this specificity, the design of these domains is governed by the principle of orthogonality. In orthogonal systems, each domain binds only with its complement, minimizing unintended spurious interactions or cross-talk with non-complementary domains. For instance, species that exhibit high cross-talk may obstruct toehold accessibility, significantly impeding reaction kinetics. Tools such as Nupack are instrumental in optimizing sequence design to reduce cross-talk[20].

2.3 Complexes and Microstates

Complexes are assemblies of interacting DNA strands. Microstates are defined by the secondary structures that form within a complex, characterized by the specific bindings between nucleotides of the complex[10, 17].

2.4 Macrostates

Macrostates describe the collective configurations that DNA complexes can adopt within a system. They represent functionally or structurally similar groupings of microstates at the domain level. This coarse-grained perspective abstracts the sequence-level details to a higher-level domain representation, facilitating the analysis and design of DNA-based systems[10, 20, 2, 1].

2.5 Reaction Rates

Reaction rates are used to profile how a system transitions between different macrostates, allowing for the prediction of system behavior over time. The rate at which two reactants initially bind, the collision rate constant, is denoted as k_1. The unimolecular reaction rate constant, k_2, describes reactions occurring within a single complex after the initial binding. These rates are combined to form the bimolecular reaction rates between two reactants via a chemical reaction network (CRN)[1, 2, 3].

2.6 Toehold Mediated Strand Displacement

Toehold mediated strand displacement is fundamental in enzyme-free DNA computation, utilizing toehold and migration domains to facilitate complex reactions. Strand displacement reactions only progress when the toeholds are complementary. Figure 2.1 illustrates the reactants, intermediates, and products of a strand displacement reaction. Figure 2.2 depicts the expected state of the system based on matching or mismatching toehold domains, showcasing toehold specificity in toehold mediated strand displacement reactions.

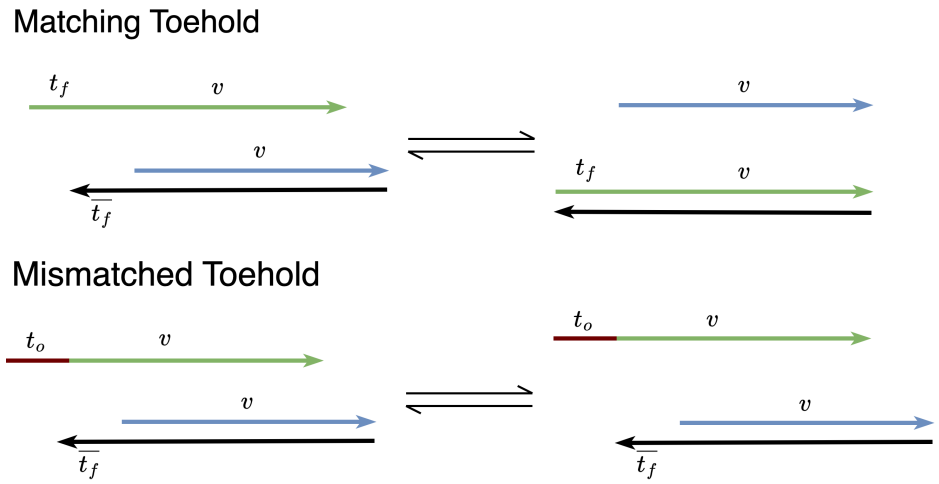


FIGURE 2.2: The strand displacement reaction will not progress if there is a mismatched toehold domain. Here, t_o is not complementary to $\overline{t_f}$.

2.7 Leak Reaction

Relying solely on kinetics to prevent off-target states ignores the potential for spurious reactions. Spurious reactions typically occur when the system overcomes a thermodynamic barrier. This may be due to bindings fraying or complete dissociation occurring. Once a leak occurs, the system may favor the erroneous configuration, or kinetic and thermodynamic barriers may prevent returning to the on-target state[14].

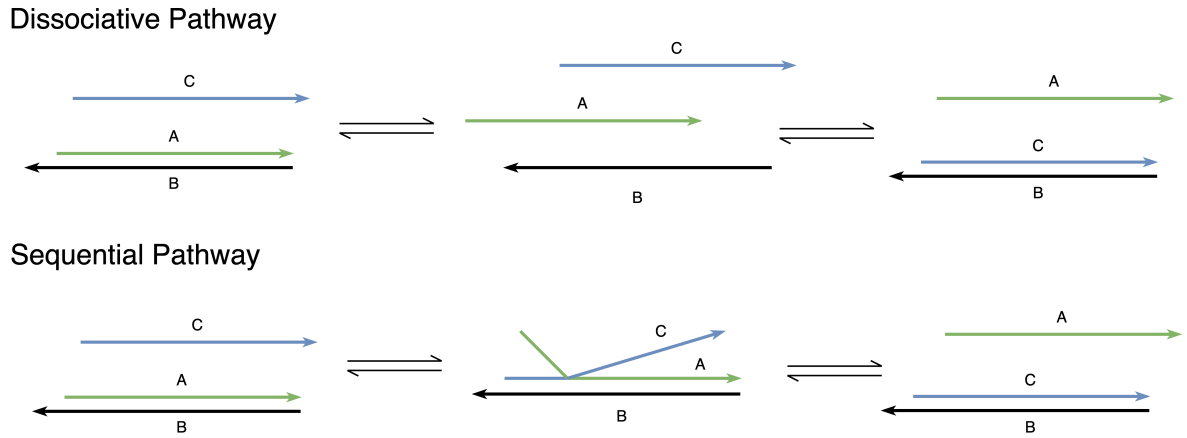


FIGURE 2.3: Leak Pathways. The incumbent strand (A) is initially bound to the substrate (B). The invader strand (C) is introduced to the solution.

These events typically occur via two main pathways:

- **Dissociative Pathway:** The incumbent strand may fully dissociate from the substrate, allowing an invading strand to erroneously bind. The likelihood of this pathway decreases at lower temperatures, but it cannot be entirely eliminated[7].
- **Sequential Displacement:** A strand initiates displacement by interacting with a partially denatured incumbent strand. This branch migration mechanism, if not properly controlled, can result in unintended toeless strand displacements[7].

Off-target states entered via a leak reaction pose significant challenges to instructions relying purely on kinetic barriers. Toehold mediated strand displacement reactions do not offer a reliable method for the incumbent strand to return to the substrate. This can lead to a buildup of off-target states that introduce errors into the computation. This underscores

the importance of not only minimizing the probability of leaks but also incorporating robust mechanisms for recovery from such events.

2.8 Cooperative Hybridization

Cooperative hybridization is a mechanism where the binding of two invading strands is required to displace an incumbent strand, effectively functioning as an AND gate in DNA-based computation[21, 13]. In this process, the presence of two specific input strands is necessary to initiate the displacement reaction.

Figure 2.4 illustrates the basic concept of cooperative hybridization. The invading strand displaces the incumbent strand from the substrate through two possible pathways. The presence of a single invading strand is insufficient to displace the incumbent strand, but the simultaneous binding of both invading strands displaces the incumbent strand[21].

2.9 Dynamic Storage using DNA in Literature

2.9.1 Stickers Model

The Sticker-Based Model is one of the earliest models designed for DNA computation. It utilizes a backbone structure with domain registers for binary data encoding, akin to traditional data storage mechanisms. On and off bits are represented by the presence or lack of top strands at a location on the backbone. Scaling up this model presents significant challenges, particularly in maintaining orthogonality as the number of registers increases. Specifically, partial bindings of top strands to the backbone can cause inaccuracies in the value stored within the system[8].

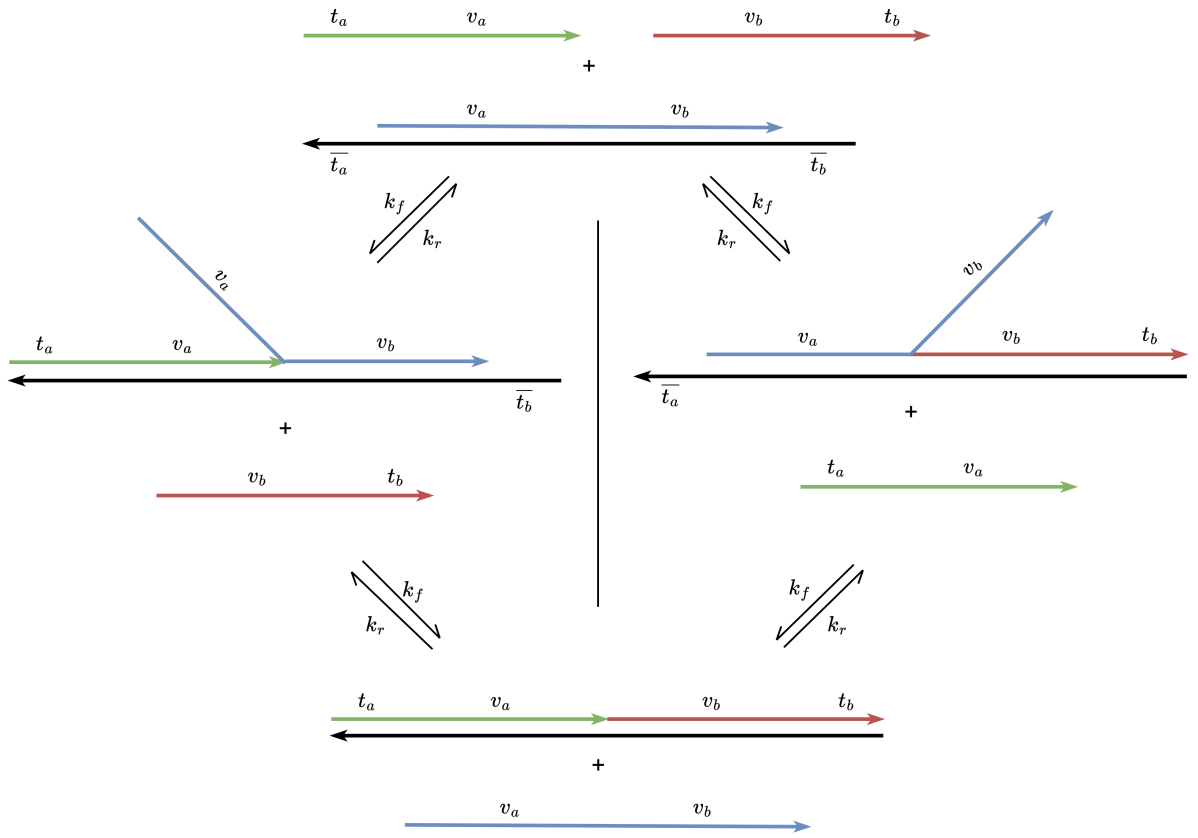


FIGURE 2.4: The reaction will only reach completion when both input strands are present.

2.9.2 SIMD DNA Structure

"SIMD DNA" is a more recent DNA computation model. This model differs from the Sticker-Based Model in that bits are encoded into the pattern of nicks of a DNA complex, minimizing open domains to prevent partial bindings. This nick structure allows deterministic, in-memory computation between adjacent registers.

Recent experimental results highlight the performance and practical constraints of the SIMD DNA architecture[15]. While enabling highly parallel information processing, SIMD DNA structures confront significant challenges, particularly error accumulation. Primary sources of these errors include losses from washing processes, incomplete reactions, and the build-up of leaks. Though advancements in washing techniques are anticipated, the yield

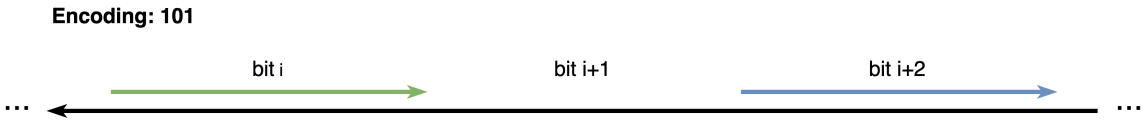


FIGURE 2.5: Encoding data using the Stickers architecture, demonstrating one of the first DNA computation and storage designs.

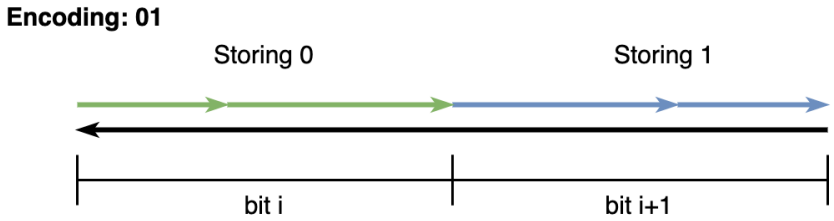


FIGURE 2.6: Encoding data using the SIMD DNA architecture, where the nick location represents a 0, or 1.

remains tightly coupled to the reaction kinetics of the design. Reliance on cooperative hybridization[1] and equal-length toehold exchanges currently poses limitations that prevent the system from functioning at low concentrations[21, 22].

Furthermore, scalability is tightly coupled to the concentration differential between instruction strands and the backbone. Higher concentrations of instruction strands, while beneficial for increased registers, simultaneously lead to spurious off-target interactions, thus magnifying leak challenges.

The authors recommend investigating hybrid computational models that integrate the strengths of SIMD DNA with other computational paradigms to enhance system performance[15].

[1] As a forward reaction mechanism.

Chapter 3

Methodology

3.1 Mitigating Leaks Through Toehold Exchange

Various strategies have been devised to mitigate leaks in DNA-based circuits. Clamps, for instance, are used in translator circuits to prevent the fraying of complex ends, acting as a physical barrier against leaks[6]. Leakless implementations extend this approach by manipulating the energy landscapes of DNA reactions to penalize spurious reporter activation.

The equilibrium concentration of a system serves as an upper limit on potential leaks, enhancing the system's leak resistance[14]. By establishing a reversible pathway from any off-target state, the system's equilibrium can be effectively biased towards the on-target state, thereby reducing leaks[14]. A detailed analysis of identifying and profiling leak states is outlined in Appendix B.5.

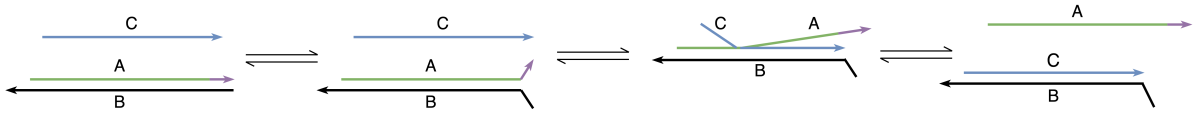


FIGURE 3.1: Clamps used in traditional strand displacement. The extra bases on the right need to fray before the invader can bind to the substrate, triggering the leak.

We propose creating the reversible pathway by extending the clamp's length from the conventional 1-3 bases to 6-10 bases, enabling it to function as a toehold instead. This modification transforms traditional clamped strand displacement into toehold exchange.

3.1.1 Toehold Exchange

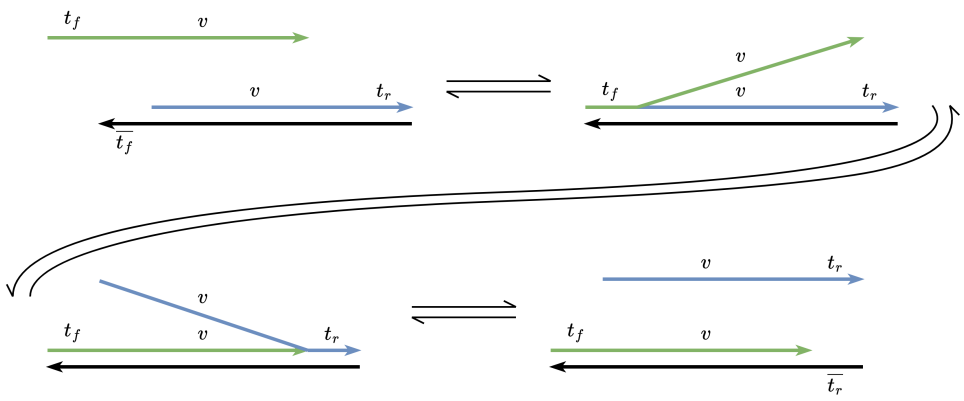


FIGURE 3.2: Toehold Exchange Mechanism. The incumbent strand competes with the invader to bind at the substrate.

Toehold exchange is DNA reaction where the invading and incumbent strand compete for binding with a substrate strand. In Figure 3.2, the forward reaction pathway is initiated by binding at the forward toehold (t_f), while the reverse pathway is initiated by binding at the reverse toehold (t_r). The migration domain is labeled as v. The lengths of t_f and t_r are denoted by n and m, respectively. The kinetics of the reaction are primarily influenced by the differnce in lengths between t_f and t_r, denoted as $n - m$[22].

Under equilbrium conditions, The influence of $n - m$ on the relative rates of k_f and k_r is quantified by analyzing the equilibrium concentration of the forward product. Figure 3.3 shows the equilibrium concentration as a function of $n - m$, calculated using bimolecular rate estimates derived from previous research[1]. This analysis confirms that for a sufficiently large $n - m$, k_f dominates k_r, reaching reaction completion. We denote the critical value of $n - m$ that ensures the desired completion rate as L.

3.1.2 Entropy Threshold

The entropy of a system at equilibrium determines the energy barrier required for stable bindings. One unit of entropy is:

[1]The bimolecular rate model assumes that the forward and reverse toehold domains consist of a mixture of G/C/A/T bases at 25°C[22]. The simplified model is invariant to input concentration.

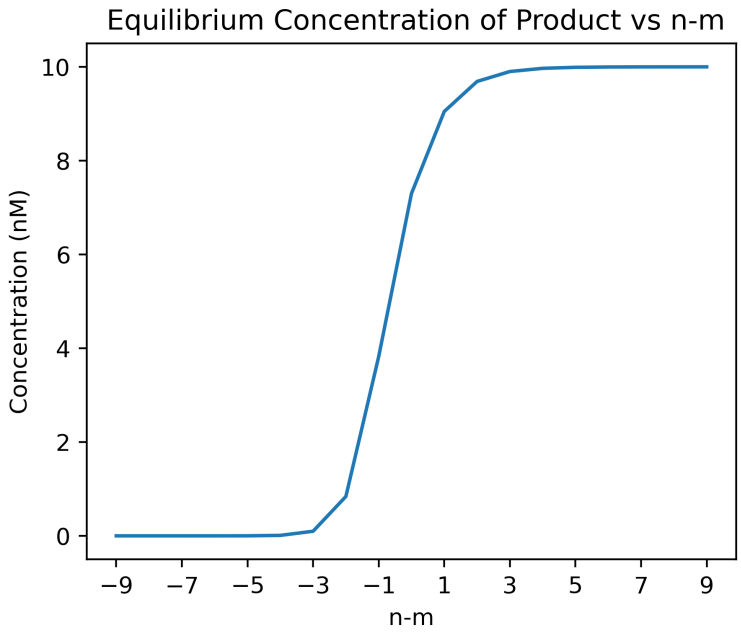


FIGURE 3.3: Equilibrium product concentration as a function of $n - m$ at 25°C[22]. Intitial reactants at 10 nM.

$$\Delta G_{assoc} + RT \ln(1/c) \tag{3.1}$$

where ΔG_{assoc} is the Gibbs free energy associated with the binding, R is the gas constant, T is the temperature in Kelvin, and c is the concentration in molar units[14, 4].

To estimate the number of nucleotide bindings needed to overcome entropy, we can use the average binding energy of a nucleotide[9]. For instance, at a temperature of 25°C, the average binding energy is 1.68 kcal/mol. At a concentration of 350 nM, approximately 7 nucleotide bindings are required to overcome the entropy barrier. For a given temperature and concentration, we denote the required number of bindings as E.

3.1.3 Leak Return Pathway

The system may enter a leak state when the invading strand lacks a forward toehold but possesses a matching migration domain (v). By employing toehold exchange mechanisms for forward reactions, we introduce a new pathway to correct such errors through strand

displacement mechanisms. It is assumed that the rate of strand displacement significantly exceeds the leak rate[7, 12].

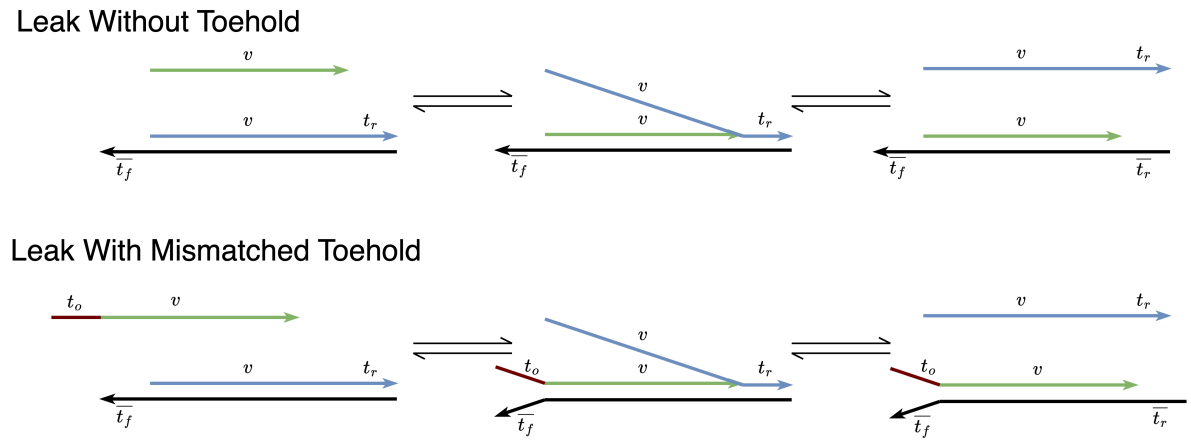


FIGURE 3.4: Illustration of the potential leak pathway during toehold exchange. The t_o toehold is not complementary to the $\overline{t_f}$ forward toehold.

When the system enters a leak state, the incumbent strand becomes unbound, and the reverse toehold on the substrate becomes accessible. The incumbent strand then rebinds via the reverse toehold, initiating strand displacement that removes the incorrectly bound invading strand. Given that the strand displacement reaction rate significantly exceeds the leak rate, the overall system at equilibrium remains primarily in on-target states. Figure 3.4 details the reaction pathways of the toehold exchange.

3.2 Storage Model

3.2.1 Encoding Data

Data is encoded into attached upper strands, referred to as *regs*. As illustrated in Figure 3.5, *regs* are comprised of two domains: δ and *addr*. The *addr* domain is further divided into subdomains ϵ, ρ and γ that are important for controlling toehold exchange. To enable computation between registers, the δ can assume either a δ_1 or δ_0, corresponding to binary on and off states, respectively. The δ_1 domain and $\overline{\delta_0}$ are designed orthogonally such that there is no binding potential, as is the δ_0 and $\overline{\delta_1}$ domains. Complementary domains are denoted

with an overline. The *addr* domains uniquely identifies the corresponding location that the data belongs to. Therefore, like traditional silicon storage, each *reg* has an address and stored value.

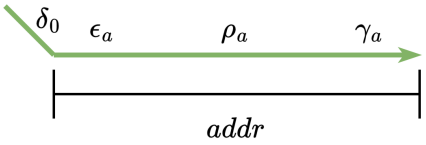


FIGURE 3.5: Domains on a *reg*. The subdomains under *addr* are shown. This *reg* is storing a zero value denoted by δ_0

3.2.2 Structure

To provide structure for the *regs*, we utilize a long, single-stranded *backbone* composed of complementary $\overline{addr}$ domains. This backbone is essential for organizing the strands, allowing for selective retrieval of *reg* via washing cycles, enabling random access memory.

Owing to the complementary nature of DNA, the *addr* domain on a *reg* will specifically bind to its counterpart on the backbone. This interaction ensures precise alignment and positioning of data strands along the backbone. The δ domain, serving as an internal toehold, extends from the bound complex, enabling subsequent toehold exchange reactions.

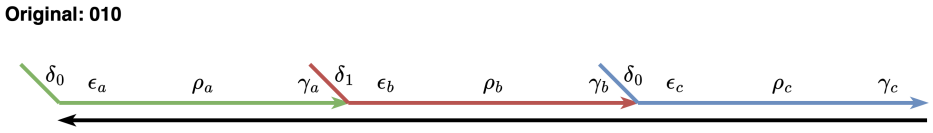


FIGURE 3.6: Bit representation of 010 in the encoding scheme. The backbone provides structure to the *regs*

3.2.3 Constraints

The domains of the *reg* strand are constrained such that equilibrium concentrations of the toehold exchange reaction are biased towards the on-target completion state.

For strand retrieval operations (Sec 3.3.1), δ is the forward toehold, and γ is the reverse toehold. The relative length of these domains are constrained following Sec 3.1.1, and the length of γ is constrained following Sec 3.1.2.

$$\text{len}(\delta) - \text{len}(\gamma) > L \tag{3.2}$$

$$\text{len}(\gamma) < E \tag{3.3}$$

During adjacency detection operations (Sec 3.3.3), (γ, ρ) and (ϵ, ρ) act as forward toeholds. γ and ϵ are reverse toeholds.

$$\text{len}(\epsilon) = \text{len}(\gamma) \tag{3.4}$$

$$\text{len}(\gamma) + \text{len}(\rho) - \text{len}(\gamma) > L \tag{3.5}$$

$$\text{len}(\epsilon) + \text{len}(\rho) - \text{len}(\epsilon) > L \tag{3.6}$$

$$\text{len}(\epsilon) < E \tag{3.7}$$

Constraint 3.5 and Constraint 3.6 simplify to:

$$\text{len}(\rho) > L \tag{3.8}$$

Two additional toehold domains are reserved for pairwise computation (Sec A.1), ϕ and ψ, constrained as follow:

$$\text{len}(\phi) = \text{len}(\psi) = \text{len}(\delta) \tag{3.9}$$

3.3 Instruction Set

This section examines the strand retrieval and adjacency detection operations enabled by toehold exchange within the model. These operations are fundamental to the model's capabilities, providing the mechanisms necessary to manipulate data.

For completeness, we use δ_x as an arbitry stored value, taking on a δ_0 or δ_1. The δ_y is the opposite stored value to δ_x, where $\overline{\delta_y}$ and δ_x have no binding potential.

3.3.1 Strand Retrieval

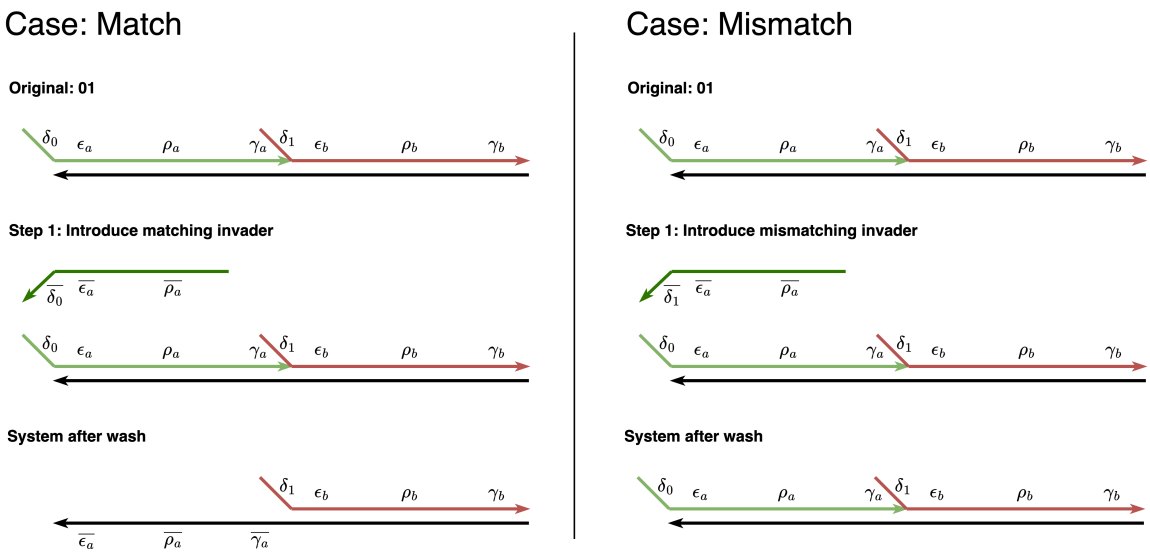


FIGURE 3.7: Overview of the match and mismatch case for strand retrieval operations.

The strand retrieval instruction is designed to selectively isolate a specific *reg* from the main solution based on its encoded *addr* and value (either δ_1 or δ_0). This selection is facilitated by a retrieval strand that is complementary to the target *addr* and δ domains.

Once the target *reg* is bound to the retrieval strand, the complex can be transferred to a separate test tube via washing for detailed analysis or further computational processing. Alternatively, if the particular value of the strand is not of interest, the wash can be discarded. This flexibility allows for both the conservation of data for future computations and elimination of unnecessary information.

We claim a retrieval strand with a $\overline{\delta_x}$ domain will only retrieve the *reg* if and only if the *reg* is storing a δ_x.

Match Case

Claim 1. *If the retrieval strand targets an arbitrary value, δ_x, and the reg stores δ_x, then reg will be retrieved from the main tube.*

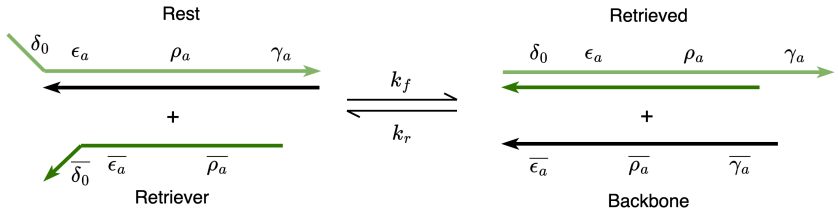


FIGURE 3.8: Strand retrieval in the match case, where a *reg* was successfully retrieved.

Proof. The retrieval strand possesses a $\overline{\delta_x}$ domain that is complementary to the δ_x domain of the *reg* strand. Binding occurs at the δ_x domain, initiating the toehold exchange. In this process, the forward toehold is δ and the reverse toehold is γ. The retrieval strand displaces the $\overline{\rho, \epsilon}$ domains of the backbone attached to the *reg*. The spontaneous dissociation of the γ domain completes the toehold exchange reaction. The system's equilbrium will be biased toward the on-target state, provided δ and γ satisfy constraints 3.2 and 3.3. Washing the system completes the retrieval process. □

Mismatch Case

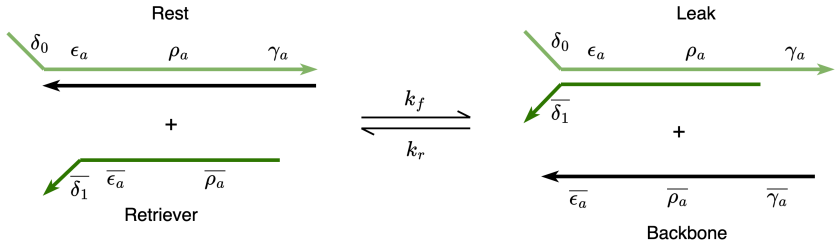


FIGURE 3.9: Strand retrieval in the mismatch case, where a *reg* has potentially leaked.

Claim 2. *If the retrieval strand targets a δ_y, and the reg stores a δ_x, then the reg will only dissociate via a leak pathway.*

Proof. The retrieval strand's $\overline{\delta_y}$ domain cannot establish a stable binding with the δ_x domain of the *reg* strand as they are not complementary. Consequently, the *reg* strand is not displaced through strand displacement or toehold exchange mechanisms. The only viable dissociation pathway for the *reg* is through a leak pathway. □

Claim 3. *If the retrieval strand is targeting a δ_y and the reg is storing a δ_x, then the reg will NOT be retrieved from the main tube.*

Proof. The *reg* may initially be displaced in the event of a leak pathway. The retrieval strand can occlude the ϵ, ρ domains of the *reg* through binding. However, the γ domain on the *reg* is able to re-bind with the open $\overline{\gamma}$ domain on the backbone. This interaction facilitates the *reg* returning to its correct position through the strand displacement pathway. As outlined in Sec 3.1.3, this reverse reaction rate significantly dominates the leak rate. Consequently, washing the system will remove the retrieval strand, leaving the *reg* bound to the backbone. □

3.3.2 Overwriting

Overwriting naturally arises from the ability to retrieve a *reg* from the backbone. Figure 3.10 details the process.

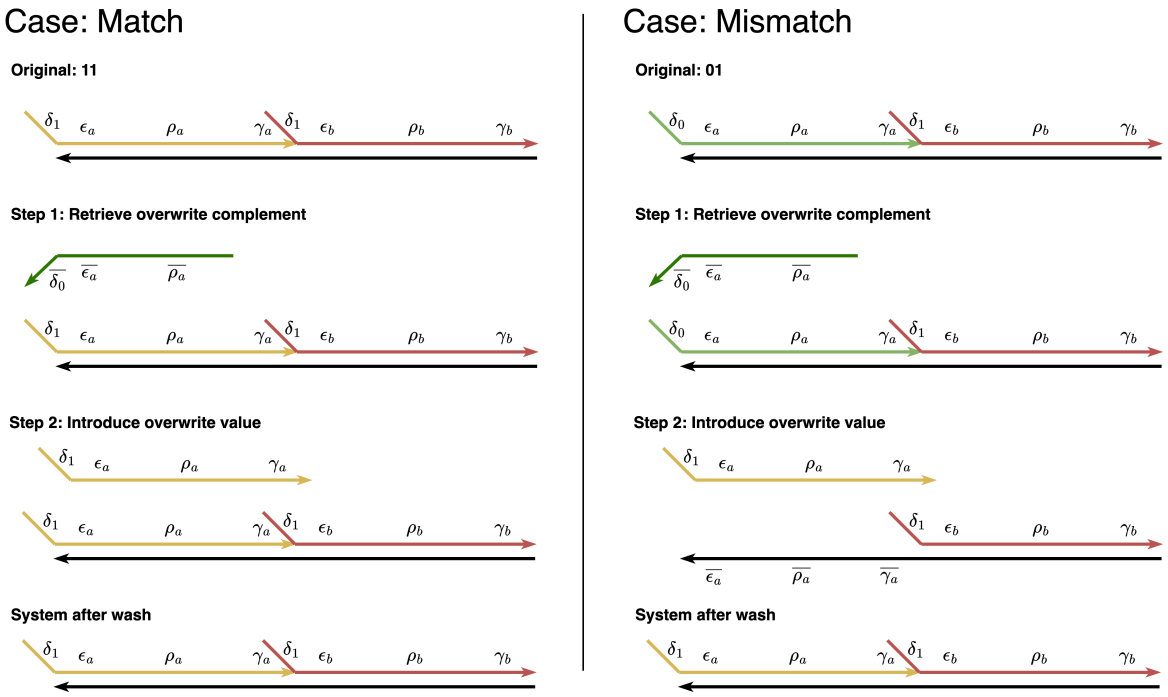


FIGURE 3.10: Overwriting values stored at an *addr*. Only one strand retrieval instruction is required. After two washing steps the old value is overwritten.

Through an overwrite operation, regardless of the previously stored value at the *reg*, a new arbitrary value δ_y is stored afterwards.

Claim 4. *If the incumbent reg stores a δ_y, a δ_y will be stored after the overwrite operation.*

Proof. A retrieval strand targeting δ_x is introduced. Due to the mismatch between the retrieval strand's target δ_x and the stored value, δ_y, the *reg* will remain attached to the backbone. Washing the system will remove the retrieval strand from the system. Subsequently, a new *reg* storing δ_y is added to the system. Since this new *reg* stores a value matching the incumbent *reg*, the strands are considered equivalent. Despite potential leak interactions, since each *reg* is equivalent, the system retains the δ_y value after washing. □

Claim 5. *If the incumbent reg stores δ_x, δ_y will be stored after the operation.*

Proof. Upon introducing a retrieval strand targeting δ_x, the system is washed, leaving the backbone's $\overline{addr}$ domain open. A replacement *reg* storing δ_y is then added. Due to the open $\overline{addr}$, this *reg* binds to the backbone. Following another wash, the *reg* at the *addr* location stores a δ_y. □

3.3.3 Adjacency Detection

Adjacency detection, a core operation in the SIMD DNA computing framework, compares values across two registers. It proceeds only when both values match a predetermined target.

Cooperative hybridization is the basis for adjacency detection[21]. The cover strand can also be used to protect the backbone when introducing replacement values in pairwise computation, outlined in Appendix A.1.

We denote the left incumbent strand as reg_a, and the right as reg_b. If neither reg_a nor reg_b are present, the cover strand binds with the backbone through complementary domains.

We claim both reg_a and reg_b remain bound to the backbone if and only if both *regs* are present at the start of the cooperative hybridization reaction.

Claim 6. *If only reg_b is bound to the backone, the cover strand will displace it.*

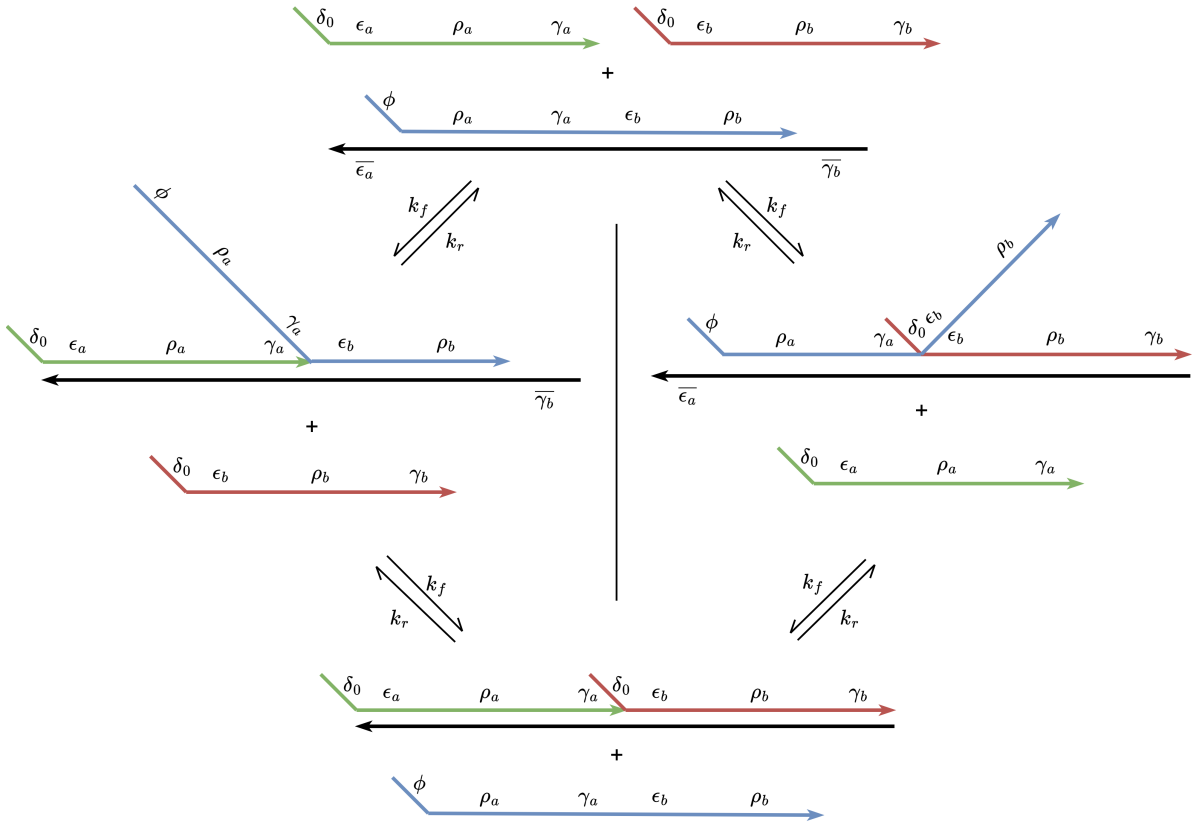


FIGURE 3.11: The Cooperative Hybridization reaction in the context of our model.

Proof. reg_b is bound to the backbone, but not reg_a. The cover strand is introduced to the system. The cover will bind to the backbone at ρ_a, γ_a. This binding initiates toehold exchange, where the forward domain is ρ_a, γ_a, and the reverse toehold is γ_b. The cover strand will dissociate the ϵ_b, ρ_b domains, with γ_b spontaneously dissociating. Following constraints 3.8 and 3.3, the forward reaction is biased at equilbrium, such that the cover strand remains bound. reg_b is removed from the system via washing. □

Claim 7. *If only reg_a is bound to the backone, the cover strand will displace it.*

Proof. reg_a is bound to the backbone, but not reg_b. By symmetry from Claim 6, the cover strand will bind at, ϵ_b, ρ_b, displacing reg_a. Washing the system will remove reg_a. □

Claim 8. *If reg_a and reg_b are present on the backbone, they remain after the adjacency operation.*

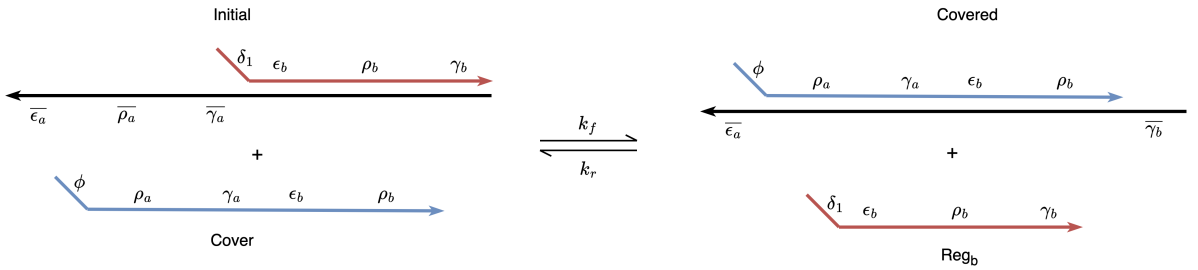


FIGURE 3.12: Displacement of reg_b by the cover strand when reg_a is not present.

Proof. reg_a and reg_b start attached to the backbone. A cover strand is introduced into the system to initiate the pairwise computation. Since the cover strand has no potential binding location with either *reg* or backbone, the cover strand is removed from the system following a wash. □

Claim 9. *If reg_a and reg_b have leaked, they will return to the backbone.*

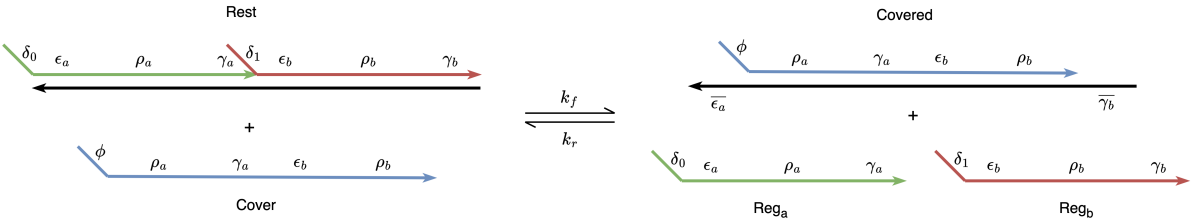


FIGURE 3.13: The system incorrectly entering a leak state. The ϵ_a and γ_b domains can initiate cooperative hybridization to return the system back to the on-target state.

Proof. The cover strand is introduced to the system. Through a leak pathway, reg_a and reg_b have both fallen off of the backbone. The cover strand, matching the $\rho_a, \gamma_a, \epsilon_b, \rho_b$ domains, binds with the backbone. reg_a is able to bind to the ϵ_a domain of the backbone (left pathway). Similarly, reg_b is able to bind to the γ_b domain. Through the cooperative hybdridization pathways detailed in Figure 3.11, the *regs* will displace the cover strand. This operation can only complete if both *regs* are present, as the left and right intermediate complexes are unstable (Claim 6, 7). □

Chapter 4

Simulation Setup

4.1 Sequence Design

The completion rate threshold was selected at 99% for 25°C operations, such that the forward bias threshold is $L = 4$.

The Nupack software was utilized to generate sequences for the simulations. Its ability to design complex orthogonal systems with minimized crosstalk renders Nupack instrumental in studying DNA architectures under varied conditions[15, 12, 10]. During the sequence design phase, system defects threshold comprises both concentration and complex defects. The target defect rate for the design step was set at 5%, encompassing both types of defects.

4.2 Thermodynamic Simulation

It is essential to ensure that at equilibrium, each instruction's minimum free energy (MFE) state aligns with the intended on-target configuration, such that forward reactions run to completion, and leak reactions remain low. Using concentrations predicted by Nupack's Analysis tool, we can estimate an upper limit on error accumulation within the system. Establishing an error threshold is vital because, at higher concentrations and temperatures, kinetic barriers that typically prevent errors begin to break down[7]. Systems are often limited to a narrow set of conditions under which they can operate reliably.

Furthermore, Nupack facilitates the exploration of potential secondary structures within the complexes, which can significantly impact reaction kinetics. Inadequate design of toeholds may lead to toehold occlusion, severely hindering the reactions. Ensuring that potential secondary structures are minimized and that toeholds remain unobstructed is a vital step before proceeding to kinetic simulations.

4.3 Kinetic Simulation

4.3.1 Peppercorn Enumeration

Peppercorn is utilized to model the system's behavior at a high level by enumerating all potential configurations of a set of input strands. It then calculates the reaction rates between these macrostates, merging intermediate macrostates into stable groupings. This step simplifies complex reaction pathways into manageable bimolecular rate estimates between resting macrostates. This abstraction reduces the kinetic complexity needed to profile the entire system into a few key bimolecular reactions[1].

4.3.2 Multistrand Simulation

Multistrand is a kinetic simulator that facilitates the interaction of multiple DNA strands simultaneously at the sequence level. Using the Arrhenius model, it offers detailed parameterization of loop interactions within complexes[23]. This provides a more precise modeling approach compared to traditional unimolecular or bimolecular rate assumptions. Multistrand is essential for simulating the initial hybridization of complexes (k_1) and the subsequent unimolecular reactions (k_2) that lead to the next macrostate[10].

4.3.3 KinDA Simulation

KinDA automates the kinetic simulation of reaction rates between stable macrostates, enumerated by Peppercorn. Using kinetic Monte Carlo samples generated by Multistrand, KinDA computes maximum a-posteriori estimates of the reaction rates k_1 and k_2[2]. The bimolecular rates estimated by KinDA are subsequently input into GPAC, an ordinary differential

equation (ODE) solver, to model the system's dynamics over time[3]. Reactions should complete within a 10-30 min timeframe at concentrations of 350 nM to be comparable with "SIMD DNA".

4.4 Profiling Leak Rates

Profiling leak reactions rates via simulation is challenging due to extended simulation duration and varied completion times. Consequently, our focus will not be on quantifying the direct rates of these leak reactions; instead, we will concentrate on profiling the recovery from the closest leak state back to the desired on-target state.

Using Nupack's predictions for equilibrium concentrations alongside KinDA's estimations of forward reaction rates, we can derive the reverse reaction rates for our system (Sec B.5 for more details).

4.5 Toehold Exchange Study

Temperature studies were conducted to understand the stability of the toehold exchange mechanism under various operational conditions. Simulations were performed at multiple temperatures: 25°C, 30°C, and 37°C.

To further explore the influence of concentration on system behavior, simulations were also conducted at three different concentrations: 10 nM, 350 nM, and 1 μM.

4.5.1 Sequences

Two sequences were optimized for different operating temperatures. Both sequences have identical domain lengths and share a matching ρ domain. `opt25` was optimized for 25°C, while `opt37` was optimized for 37°C. These sequences evaluate the impact of toehold composition on the leak rate, independent of domain length. The γ domain of `opt37` exhibits a stronger binding energy of -6.67 kcal/mol, compared to -5.20 kcal/mol for the γ domain of `opt25` [1].

[1]Calculated using simulation parameter B.1 in Nupack at 37°C

Additionally, a sequence was designed based on `opt25` but with the δ, ϵ, and γ domains increased by a single base, referred to as `opt11_7`. This adjustment resulted in a new binding energy of -6.59 kcal/mol for the γ domain.

While `opt25` was subjected to both thermodynamic and kinetic analysis, `opt37` and `opt11_7` were evaluated only through thermodynamic simulations.

4.6 Simulating Instructions

The reactions outlined in Section 3.3 serve as the basic units for each operation within our model. Pairwise operations, as detailed in Appendix A.1, are also included.

- Strand Retrieval3.3.1
  - Match Case
  - Mismatch Case
- Adjacency Detection3.3.3
  - Match Case
  - Mismatch Case
- Pairwise Computation
  - Cover RetrievalA.1.1
  - Attempted Retrieval of Cover StrandA.1.2
  - Attempted Retrieval of *regs* by Cover Retrieval StrandA.1.1

Chapter 5

Simulation Analysis

5.1 Design Analysis

The design phase achieved our target of maintaining defects below the 5% threshold. The cover strand was identified as the largest contributor to system's structural defects. Figure 5.1 illustrates the secondary structures formed in the cover strand. Toeholds for pairwise compuatation were not occluded as evident from a domain level analysis, ensuring that kinetic performance is not hindered.

5.2 Analysis of Equilibrium Concentrations

The selected value of L allows for up to 1% of the system not reaching completion. To investigate the sources of this error, the equilibrium concentrations of reactants and intermediate microstates along the intended pathway were analyzed. Figure 5.2 demonstrates that errors in completion are largely attributed to the presence of a high concentration of intermediate complexes.

The increased buildup of intermediate complexes suggests that some strands overcome the entropy barrier, remaining bound. This finding indicates that while a longer or stronger reverse toehold can suppress errors by improving binding efficiency, there is a critical balance required.

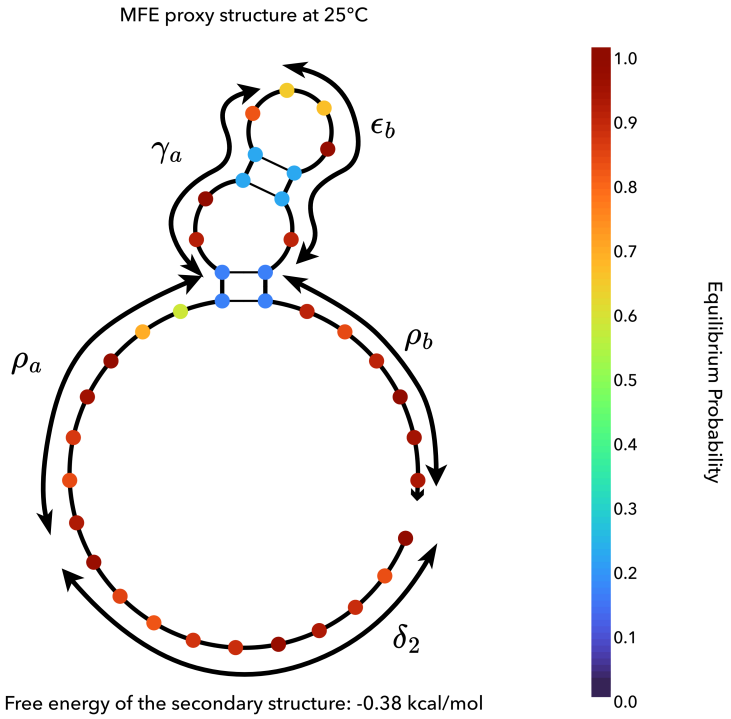


FIGURE 5.1: The equilibrium secondary structures that form within the cover strand. Domains as specified in Figure 3.13

5.3 Toehold Exchange Study

5.3.1 `opt25` Analysis

Nupack analysis confirms that the single register system for `opt25` achieves equilibrium with over 99% completion. Kinetic analysis indicates that on-target reactions reach completion within the targeted 10-30 min reaction period.

`opt25` exhibited minimal error across all concentrations. In the study conducted at 1000 nM, the error rate remained under 1% at 25°C. However, as temperatures increased, so did the leak accumulation. At 37°C, the highest leak rate observed was 2.9%. Despite these higher leak rates, the completion rate remained stable, suggesting that the forward and reverse reaction rates effectively balanced each other, even at elevated temperatures and concentrations.

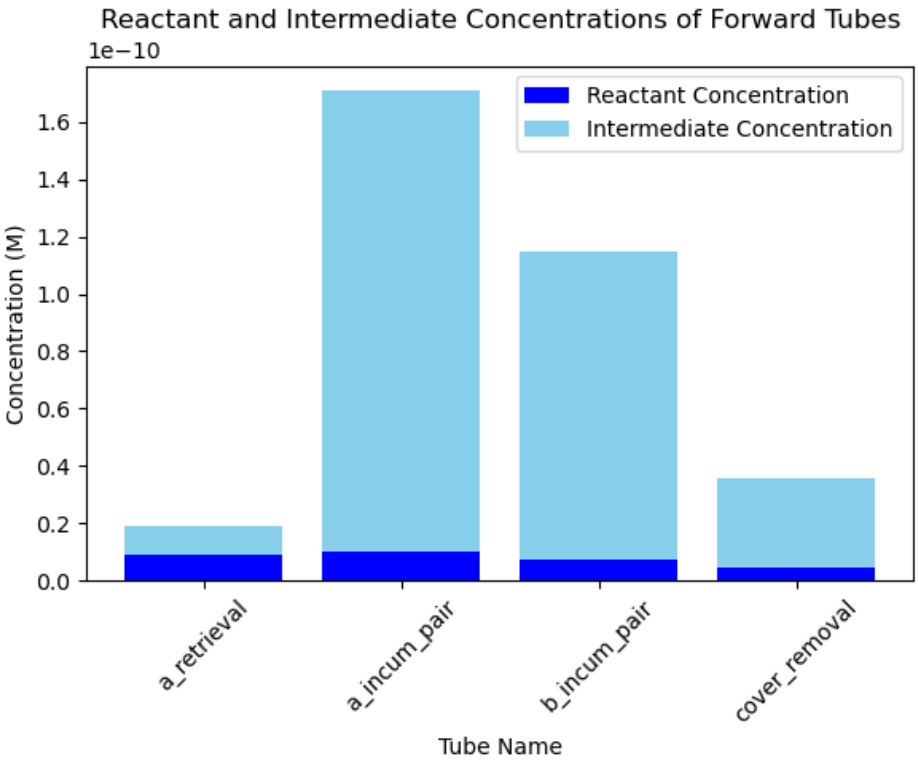


FIGURE 5.2: Intermediate complexes constitute the majority of off-target complexes.

5.3.2 Alternative Sequences

Equilibrium studies for alternative sequences were conducted at 37°C. `opt37` maintained a 99% completion rate, whereas `opt11_7` achieved 98%. Considering Sec 5.2, the longer reverse toehold may be more impactful on the dissociation rate than sequence makeup.

The results demonstrate that `opt37` is exceptionally robust at higher temperatures, exhibiting a leak rate of only 0.57% at equilibrium, emphasizing the need for stronger reverse toeholds in high-temperature environments. Similarly, the extended toehold in `opt11_7` showed a comparable enhancement in performance, with a leak potential of only 0.68% at equilibrium. These findings support the notion that both `opt37` and `opt11_7` benefit from enhanced reverse toeholds that effectively suppress leaks. However, they also highlight a crucial balance between toehold length and the accumulation of intermediate complexes—excessively long toeholds may overcome entropy barriers. This observation underscores the temperature

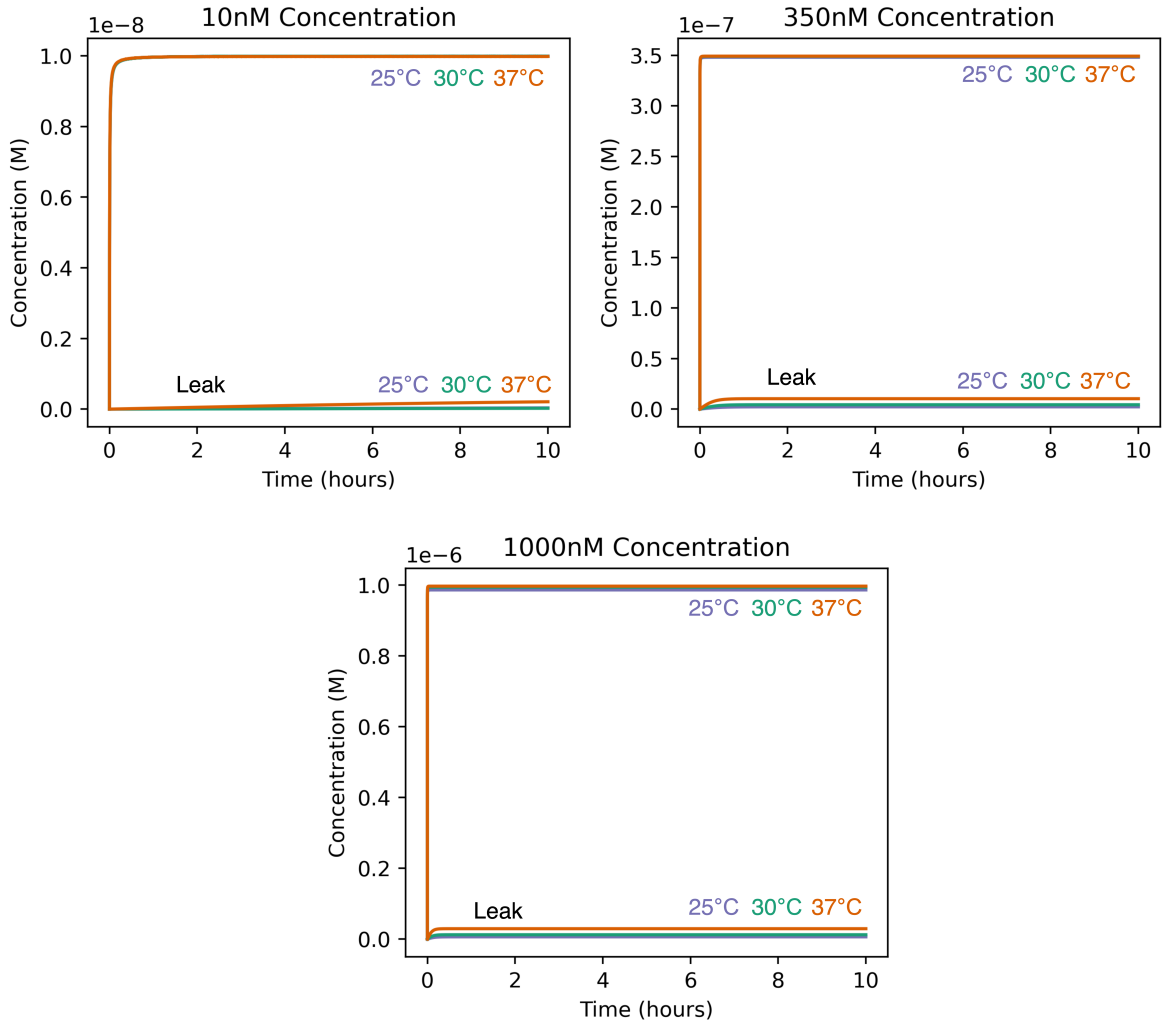


FIGURE 5.3: Reaction kinetics for the `opt25` sequence at various concentrations and temperatures.

dependency of the γ domain's binding energy, necessitating a balance between strength and toehold length to effectively suppress leak rates.

5.4 Multi-Register Results

Based on the previous design, we anticipated minimal to no leaks at 25°C, and the results corroborate this expectation. Consequently, our focus shifted to examining the completion rate of the system. It demonstrated that on-target reactions consistently reached completion

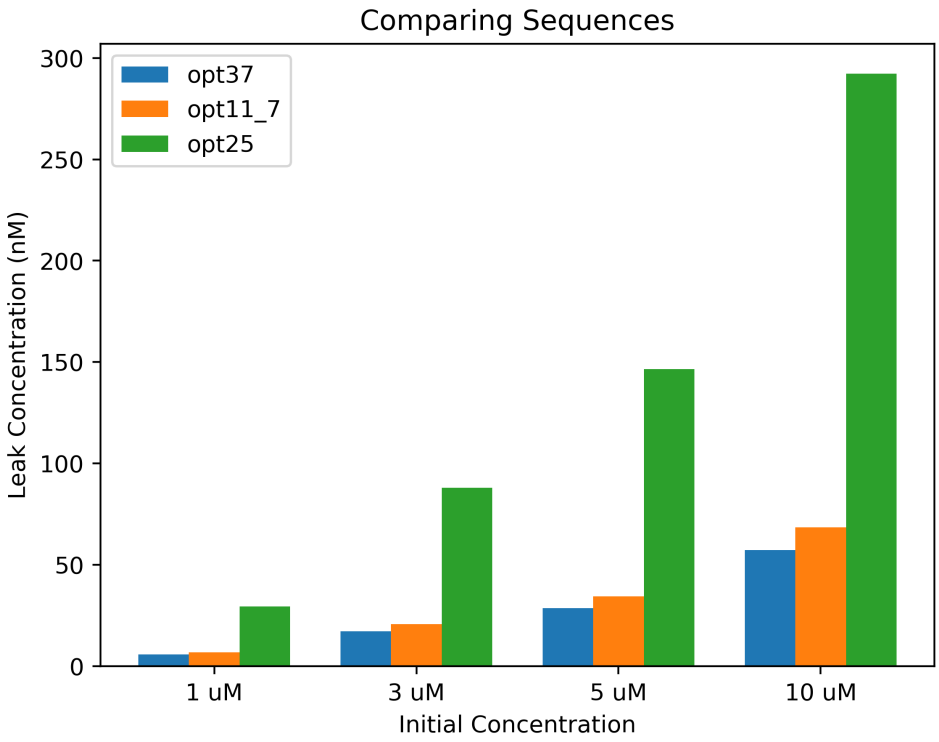


FIGURE 5.4: Comparison of sequences optimized for different operating temperatures. Equilibrium calculations at 37°C.

within the targeted 10-30 minute timeframe, underscoring the system's ability to reliably meet completion thresholds.

The most significant source of error was identified in the pair adjacency system. At 10 nM, when attempting to remove reg_a with the cover strand, the completion rate reached 98%, falling below the desired 99% threshold. As illustrated in Figure 5.2, the off-target complexes in this reaction were predominantly intermediate complexes. This issue is likely due to the extended migration domain required to displace reg_a, which promotes further intermediate complex accumulation.

At 350 nM and 37°C, the system struggled to reach completion for the same reaction. This suggests that the kinetics of this particular reaction require further exploration. A possible cause is proposed in Appendix B.4.

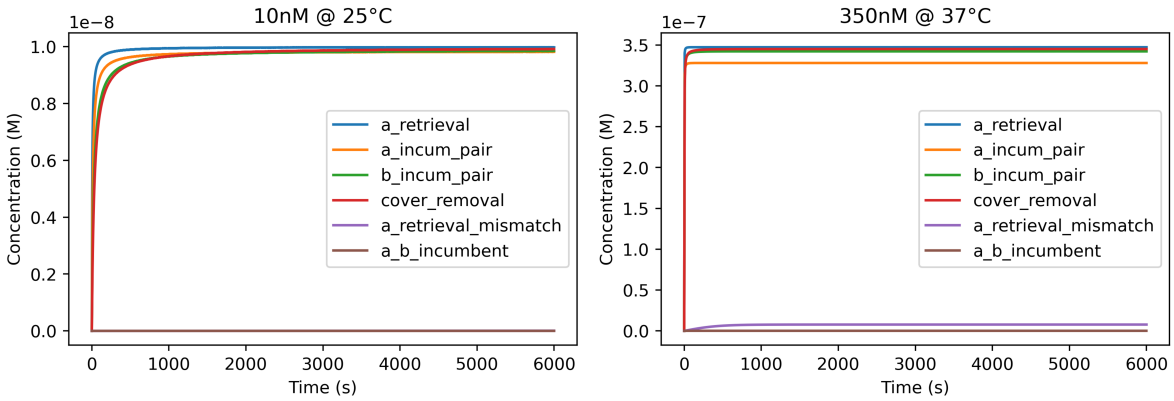


FIGURE 5.5: Initial concentrations as indicated in title. Instructions with less than .1% leak concentration at equilibrium are not displayed.

Moreover, the mismatch retrieval mirrors the results of the storage system at elevated temperatures, demonstrating consistency across the simulations. It is clear that the system length must be optimized for specific reactions and operating temperatures to function correctly.

Further studies are required to analyze the temperature dependence of the computation system. These results confirm that operations complete within a reasonable timeframe and that the system does not accumulate significant errors at target conditions.

Chapter 6

Discussion

This thesis addressed the challenge of designing a DNA-based system capable of both data modification and computational tasks. The proposed design framework adhered to a well-defined constraint system to ensure reliable operation.

Each *reg* strand in the system serves a dual purpose: acting as a unit of storage and a medium for computation. The inclusion of internal toeholds facilitates efficient RAM operations. By applying enthalpic and entropic domain length constraints, the system ensures that reactions consistently achieve the desired target states.

A significant issue addressed was the reliance on slow reaction primitives that require higher concentrations to function effectively. Our approach aimed to overcome these limitations by employing toehold exchange as the primary forward reaction. This strategy enables the system to reach completion while maintaining high performance across varying concentrations. However, the design must account for operating temperature, as increased temperatures necessitate specific adjustments to design parameters to maintain optimal performance.

Pairwise computation across adjacent registers proved largely successful. Further investigation using advanced models is still needed to understand the impact that internal dangles have on the cooperative hybridization reaction.

Future research should focus on the experimental implementation of the model. Additionally, the model can be enhanced by introducing redundancy, storing data across multiple systems to address the challenges of orthogonal design.

Finally, the constraint framework could also be expanded to incorporate energy-based

constraints into the sequence design, instead of solely relying on domain length measures. This would account for the impact domain composition has on reaction kinetics.

The model successfully addressed the outlined problem, showing potential for scalable extensions. The research demonstrated a novel approach to DNA-based computation and storage, providing a robust framework for future advancements in the field.

Appendix A

Programs

A.1 Pair Computation Program

Pairwise computation extends the principle of adjacency detection by ensuring that *regs* are overwritten only if both match the target values. This process involves using two tubes to complete the full operation; the second tube is essential for preserving the state within the main system.

We claim that strand retrieval operations in conjunction with adjacency identification allows selective pairwise computation based on stored values.

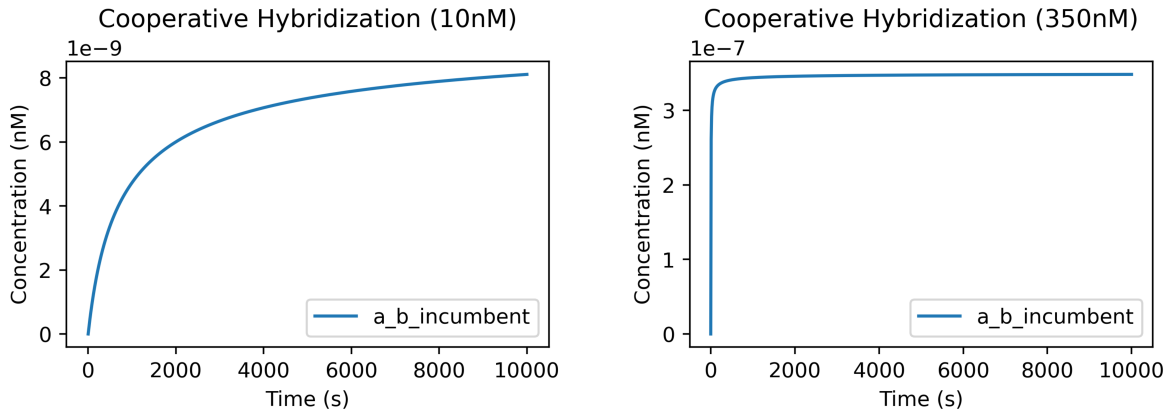


FIGURE A.1: Instruction relying on Cooperative Hybridization require higher concentrations to complete within the 10-30min time frame.

A.1.1 Cover Strand Removal

In the "SIMD DNA" model, cooperative hybridization typically functions as a forward reaction step, representing one of the slower processes within the system. However, by adding a toehold on the cover strand, cooperative hybridization primarily serves to prevent leaks, not as a forward reaction.

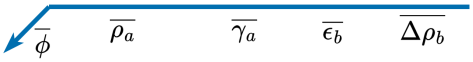


FIGURE A.2: Diagram illustrating the domains comprising the cover retrieval strand.

The cover retrieval strand contains complementary domains to those on the cover strand, enabling removal through toehold exchange. Specifically, the $\overline{\Delta\rho_b}$ domain is designed to facilitate this process. For the cover retrieval operations to reach completion and maintain consistency with previous constraints outlined in Sec 3.2.3, $\overline{\Delta\rho_b}$ is constrained:

$$\text{len}(\rho_b) - \text{len}(\overline{\Delta\rho_b}) = \gamma_b \tag{A.1}$$

$$\text{len}(\phi) - \text{len}(\rho_b) + \text{len}(\overline{\Delta\rho_b}) > L \tag{A.2}$$

$$\text{len}(\overline{\Delta\rho_b}) < E \tag{A.3}$$

Attempted Retrieval of *regs* by Cover Retrieval Strand

During pairwise computation, the cover strand is retrieved by the cover retrieval strand. This cover retrieval strand could potentially bind with *regs* if they are present. However, we claim that the cover retrieval strand will only remove the cover strand from the system.

Claim 10. *If reg_a and reg_b are bound to the backbone, the cover retrieval strand will not retrieve either of them.*

Proof. The backbone, reg_a, and reg_b do not expose any domains that are complementary to the cover retrieval strand. The cover retrieval strand is washed away, leaving reg_a and reg_b attached to the backbone. □

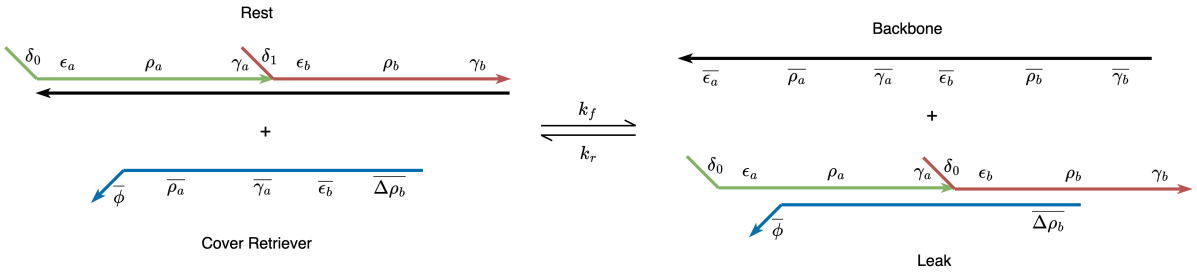


FIGURE A.3: Illustration of a leaked cover retrieval strand. The cover retrieval strand incorrectly removes the incumbent *regs* when a cover strand is not present.

Claim 11. *If reg_a or reg_b have leaked from the backbone, the cover retrieval strand will not remove them from the system.*

Proof. Should reg_a or reg_b dissociate from the backbone via a leak pathway, the cover strand could potentially occlude the ρ_a, γ_a domains of reg_a and the ϵ_b, ρ_b domains of reg_b. Despite these occlusions, the unbound ϵ_a domain of reg_a and the γ_b domain of reg_b remain free. These unbound domains can subsequently bind with their complementary domains on the backbone. Through strand displacement facilitated by the backbone, the cover retrieval strand is displaced. This ensures that both reg_a and reg_b reattach to the backbone. Washing the system removes the unbound cover retrieval strand. □

Cover Strand Retrieval

Claim 12. *The cover retrieval strand selectively removes the cover strand from the backbone.*

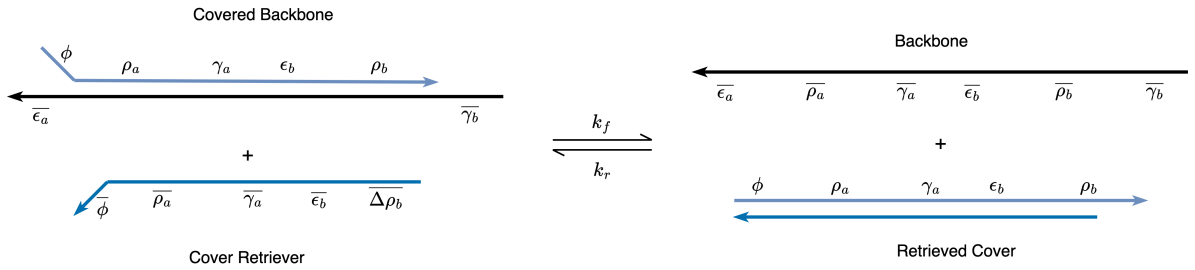


FIGURE A.4: Illustration of the cover retrieval strand removing a cover from the backbone.

Proof. Initially, the cover strand is bound to the backbone within the system. Upon introduction of the cover retrieval strand into the solution, it specifically targets and binds to the complementary ϕ domains located on the cover strand. This interaction initiates a toehold exchange mechanism, displacing the cover strand from the backbone following constraints A.2 and A.3.

Subsequent washing of the system removes the now unbound cover strand from the solution. Referencing the analysis in Sec A.1.1 and supported by this proof, we establish that the cover retrieval strand exclusively removes the cover strand, retaining *regs* within the system. □

A.1.2 Attempted Retrieval of Cover Strand

During pairwise computation, a cover strand may be bound to the backbone. Retrieval strands targeting the *regs* should not mistakenly retrieve the cover strand.

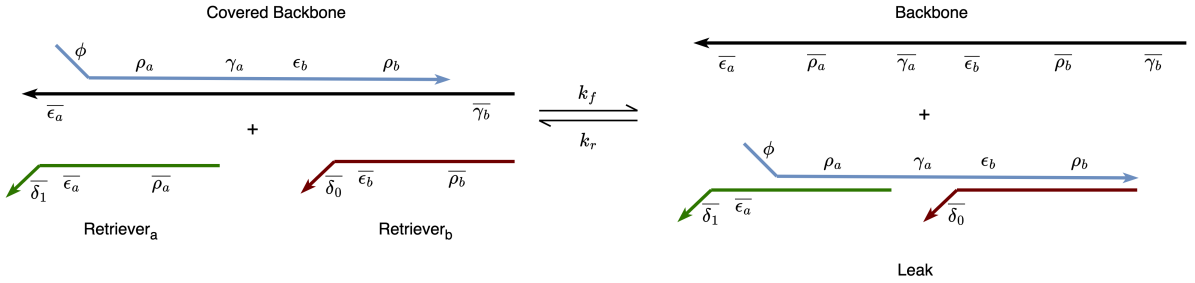


FIGURE A.5: Illustration of the cover strand dissociating via a leak pathway, with retrieval strands subsequently binding to it.

Claim 13. *Retrieval strands cannot remove a cover strand that is attached to the backbone.*

Proof. Consider a scenario where the cover strand is bound to the backbone at the $\rho_a, \gamma_a, \epsilon_a, \rho_b$ domains. When retrieval strands are introduced, there are no open domains on cover strand that are complementary. Consequently, after washing the system, the cover strand remains attached to the backbone and the retrieval strands are washed. □

Claim 14. *Retrieval strands will not retrieve a cover strand that has leaked from the backbone.*

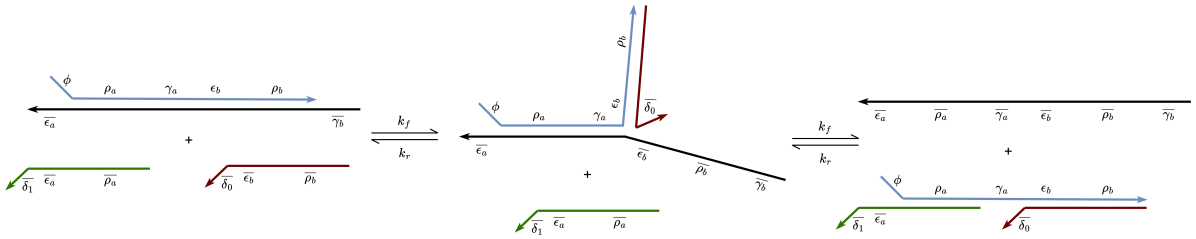


FIGURE A.6: Detailed pathway of the leaked cover strand occluded by retrieval strands.

Proof. If the cover strand dissociates from the backbone, it exposes complementary domains to the retrieval strands. While retrieval strands may occlude the $\rho_a, \epsilon_b, \rho_b$ domains of the cover strand, the γ_a domain remains unbound. This allows the cover strand to reattach to the backbone through the open γ_a domain, facilitated by sequential strand displacement reactions. Figure A.6 illustrates this return pathway. Following these reactions, the cover strand is returned to the backbone and any retrieval strands are dissociated. Washing the system subsequently removes any unbound retrieval strands. □

A.1.3 Removable Retrieval Strand

During pairwise computation, incumbent *regs* are temporarily stored in a secondary tube to preserve state within the system. To reintroduce these *regs* into the main system, they must be detached from the retrieval strands. A removable retrieval strand equipped with a $\overline{\psi}$ domain is used to facilitate this process. This design allows a complementary strand to clear the *regs* from the removable retrieval strand, effectively rendering it unreactive.

Claim 15. *The removable retrieval strand retrieves reg strands.*

Proof. The removable retrieval strand is designed with consecutive ρ, ϵ, and δ domains. The additional $\overline{\psi}$ domain extends from the δ domain. Despite this added domain, the strand effectively binds to the complementary δ domain of a targeted *reg*, initiating a toehold exchange process. This interaction leads to the removal of incumbent *regs* from the main system, analogous to the action of a standard retrieval strand. □

Claim 16. *The reg can be detached from the removable retrieval strand after washing.*

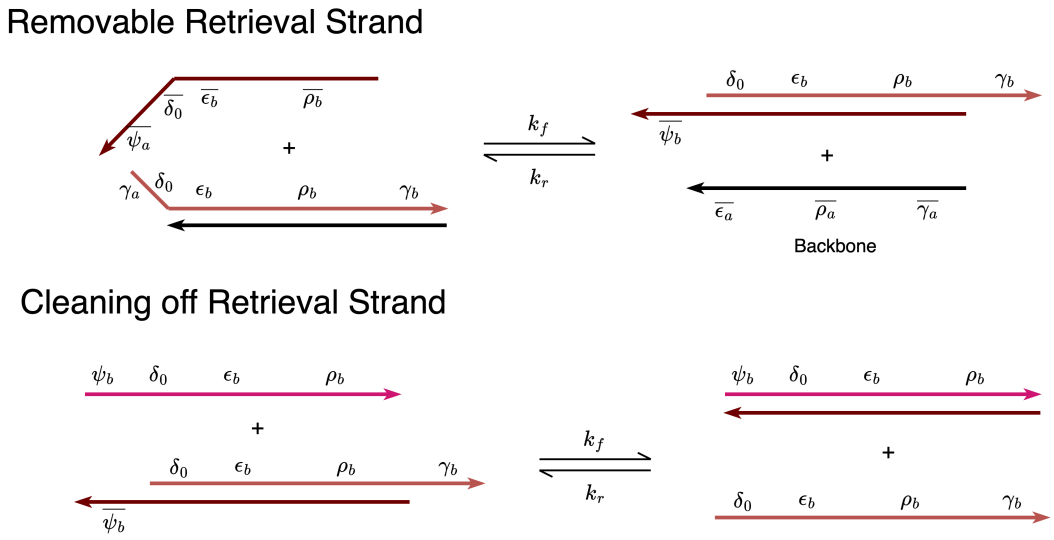


FIGURE A.7: The removable retrieval strand allows *reg* domains to be reintroduced into the main tube by cleaning them from the retrieval strand.

Proof. A new strand containing the domains $\psi, \delta, \epsilon, \rho$ is introduced into the secondary tube. The ψ domains on this new strand will bind to the corresponding $\overline{\psi}$ domain on the removable retrieval strand. This interaction initiates a strand displacement process, dissociating the incumbent *reg*. The complex has no open domains, reaching equilibrium with the *reg* fully separated from the removable retrieval strand. □

A.1.4 Complete Pairwise Program

Using strand retrieval: 3.3.1, adjacency detection 3.3.3, and removable strand retrieval A.7, we are able to create pairwise operations.

We claim that if and only if both reg_a and reg_b have the target values δ_{xa} and δ_{xb}, then the value will be overwritten with δ_{ya} and δ_{yb} respectively.

Claim 17. *If both reg_a and reg_b store their target values, they will be overwritten.*

Proof. Consider reg_a storing δ_{xa} and reg_b storing δ_{xb}, which are their respective target values.

1. **Removable Strand Retrieval:** Initially, removable retrieval strands targeting δ_{ya} and δ_{yb} are introduced for each *reg*, respectively. These retrieval strands have mismatching

toeholds with the stored values of reg_a and reg_b. A subsequent wash removes these strands from the solution, isolating the wash into a secondary tube.

2. **Adjacency Detection and Secondary Tube Operation:** This step involves simultaneous operations in the main and secondary tubes:

   (a) In the main tube, a cover strand is introduced. As both reg_a and reg_b are present, the cover strand is removed with a wash, which is also isolated in the secondary tube.

   (b) Concurrently, cleaning strands are introduced in the secondary tube to clear the *reg* from the removable retrieval strands. They bind directly without displacing a *reg*.

3. **Strand Retrieval:** Retrieval strands targeting δ_{xa} and δ_{xb} are introduced. Matching with their respective targets, reg_a and reg_b are removed from the backbone. The system is then washed, discarding the wash.

4. **Introduce Replacement for reg_a:** A new reg_a storing the δ_{ya} value is introduced and binds with the backbone. The system is wash.

5. **Introduce Replacement for reg_b:** Similarly, a new reg_b storing the δ_{yb} value is introduced and binds to the backbone. The system is wash.

6. **Retrieve Cover Strand:** A cover retrieval strand is introduced, and subsequently washed out of the solution as it does not bind with reg_a or reg_b.

7. **Return Secondary Tube:** The contents of the secondary tube are reintroduced to the main tube. The cover strand remains unbound. The cleaned removable retrieval strands contain no open domains. A final wash ensures that only reg_a storing δ_{ya} and reg_b storing δ_{yb} remain in the solution. The operation is now complete.

□

Claim 18. *If either reg_a or reg_b are not storing their target values, the system will remain the same.*

Proof. Consider reg_a storing δ_{ya} and reg_b storing δ_{xb}, where δ_{xb} is the target value, but δ_{ya} is a mismatch.

1. **Removable Strand Retrieval:** Initially, removable retrieval strands targeting δ_{ya} and δ_{yb} are introduced for each reg, respectively. reg_a is removed from the backbone as it is storing a δ_{ya}. The wash is isolating in a secondary tube.

2. **Adjacency Detection and Secondary Tube Operation:** This step involves simultaneous operations in the main and secondary tubes:

   (a) In the main tube, a cover strand is introduced. Since only reg_b is present, the cover strand displaces reg_b. reg_b is washed and isolated in the secondary tube, while the cover strand remains bound to the backbone.

   (b) Concurrently, cleaning strands are introduced in the secondary tube to clear the reg from the removable retrieval strands. reg_a is cleaned from its removable retrieval strand.

3. **Strand Retrieval:** Retrieval strands targeting δ_{xa} and δ_{xb} are introduced. As both reg_a and reg_b are removed from the main tube, these do not bind to the cover strand. The system is then washed removing the retrieval strand, discarding the wash.

4. **Introduce Replacement for reg_a:** A new reg_a storing δ_{ya} is introduced. The cover strand prevents the replacement reg_a from binding. Washing the system removes the replacement reg_a.

5. **Introduce Replacement for reg_b:** Similarly, a new reg_b storing δ_{yb} is introduced. The cover strand prevents the replacement reg_b from binding. Washing the system removes the replacement reg_b.

6. **Retrieve Cover Strand:** A cover retrieval strand is introduced. This cover retrieval strand removes the cover strand from the solution with a wash.

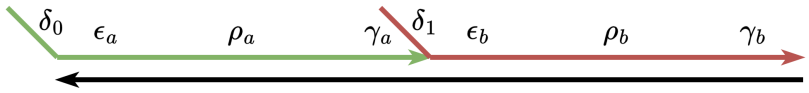
7. **Return Secondary Tube:** The contents of the secondary tube are reintroduced to the main tube. The cleaned removable retrieval strands contain no open domains. The reg_a

and reg_b return to the backbone. The operation is now complete, with the incumbent reg_a and reg_b remaining on the backbone after the mismatch.

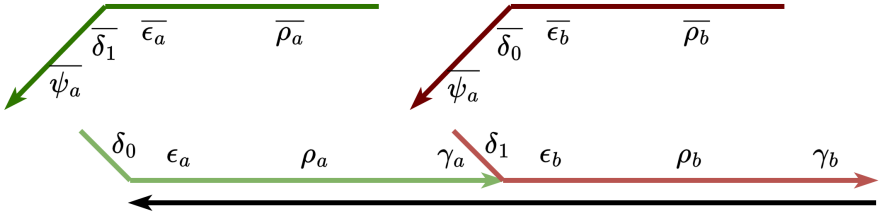
□

Case: Match

Original: 01

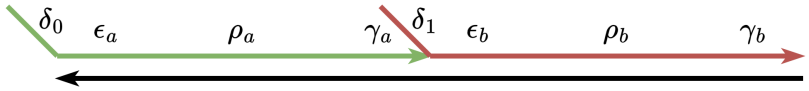


Step 1a: Introduce retrieval strands

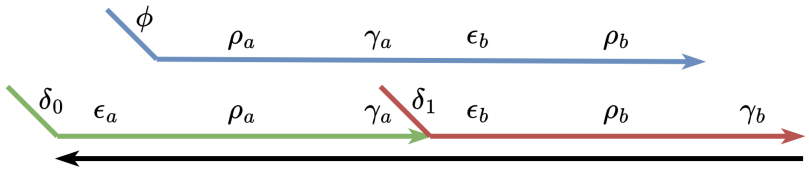


Step 1b: System After Washing

Isolate Wash

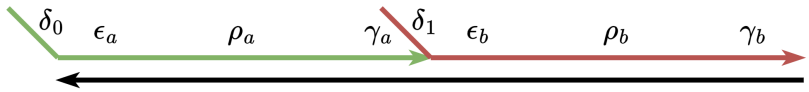


Step 2a: Cover Strand

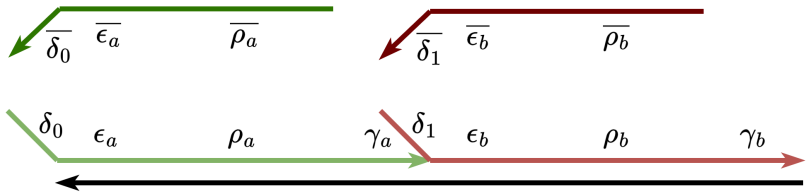


Step 2b: System after Washing

Isolate Wash



Step 3: Introduce retrieval strands with target delta



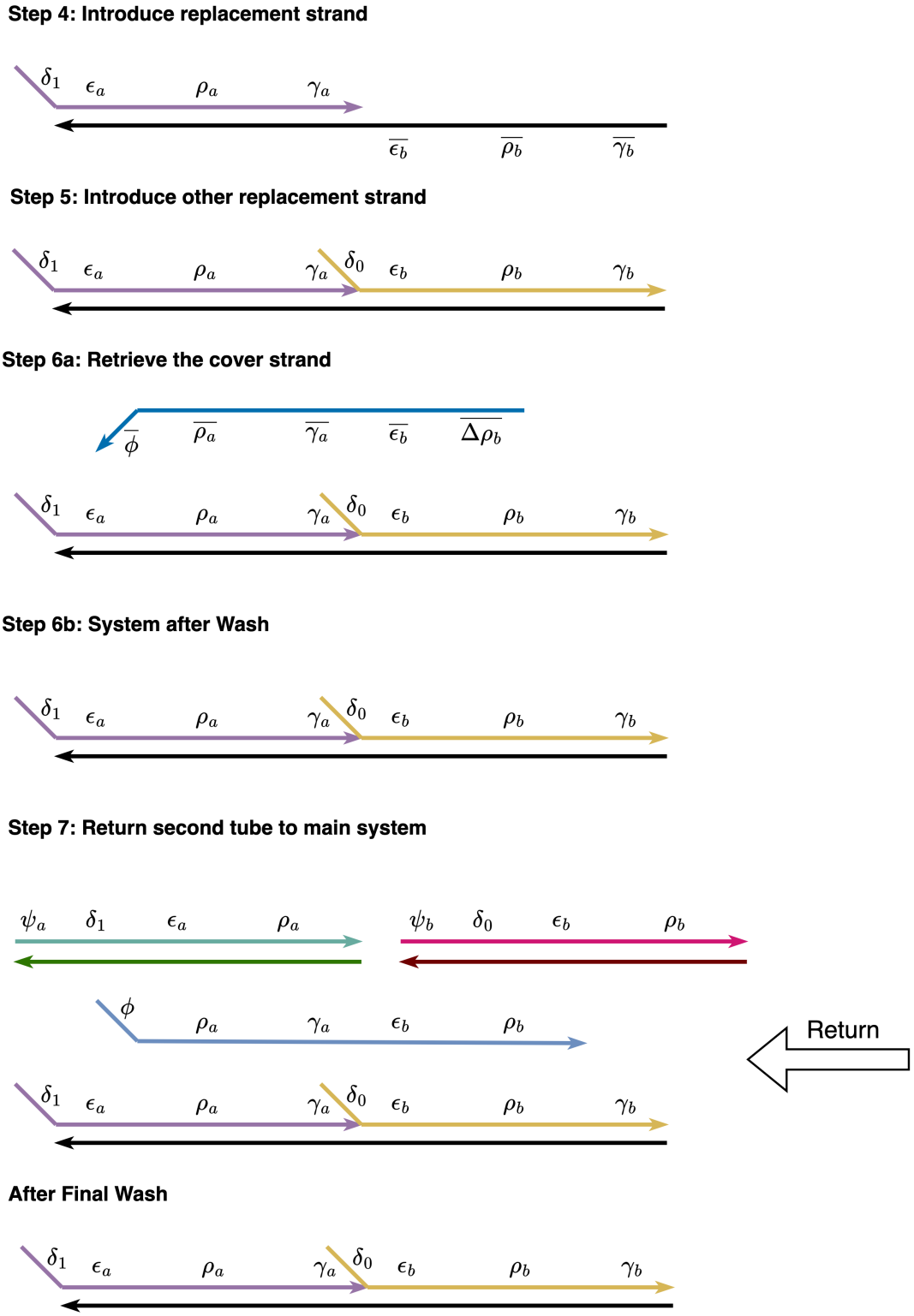


FIGURE A.8: The entire pairwise algorithm in detail. For isolation we do not discard the washed solution.

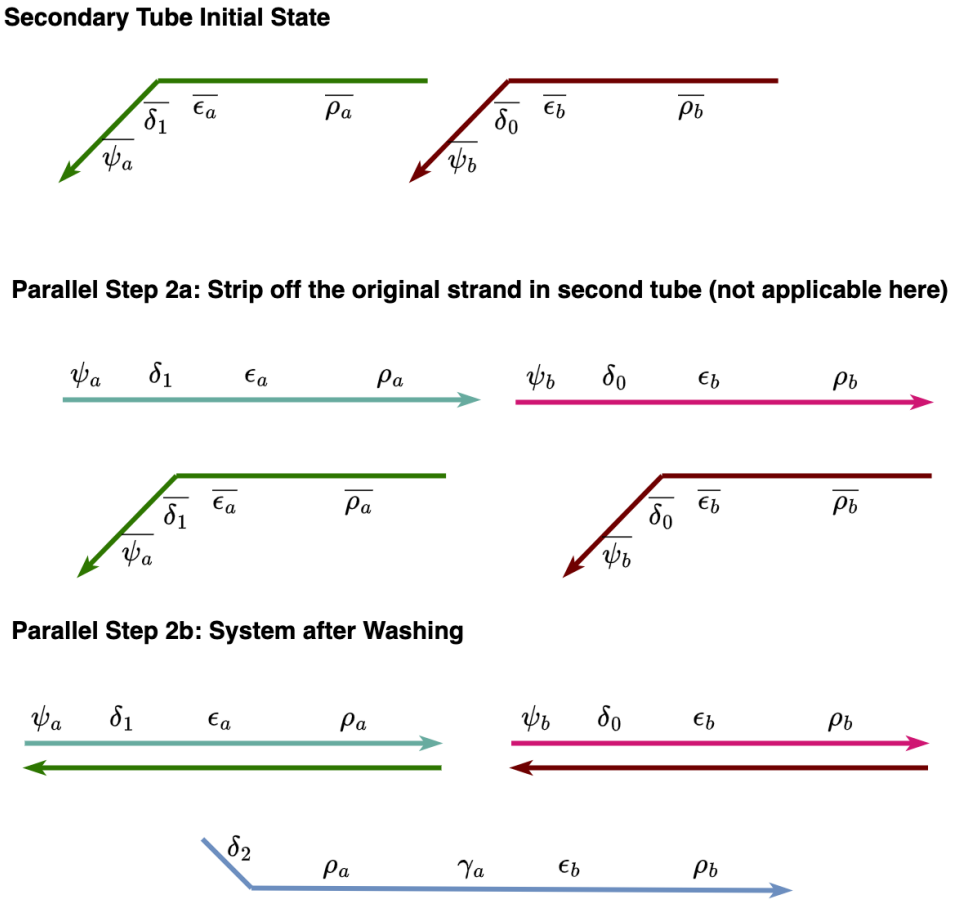
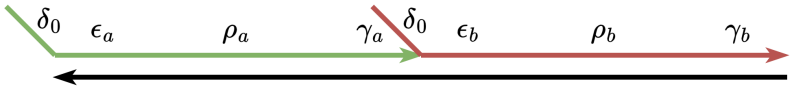


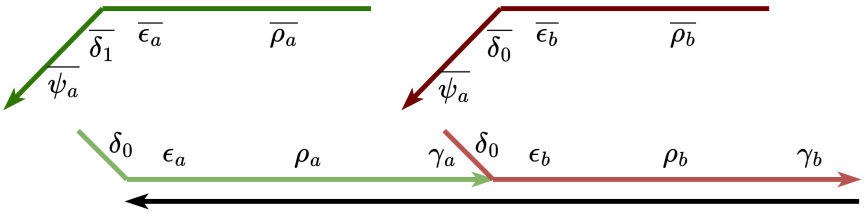
FIGURE A.9: An additional tube is required to preserve state within the system. The cover strand is introduced to the secondary tube during an isolation step.

Case: Mismatch

Original: 00

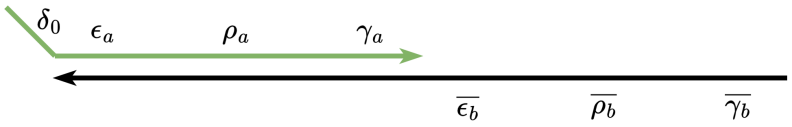


Step 1a: Introduce retrieval strands

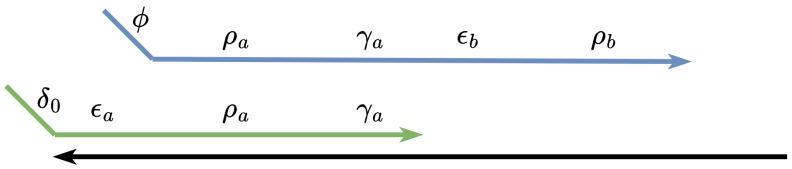


Isolate Wash

Step 1b: System After Washing

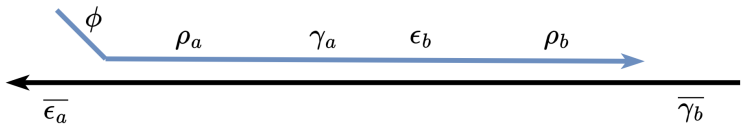


Step 2a: Cover Strand

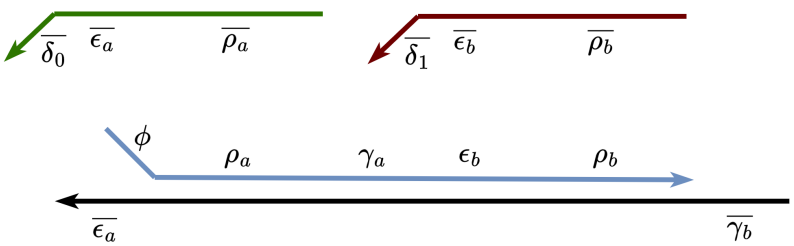


Isolate Wash

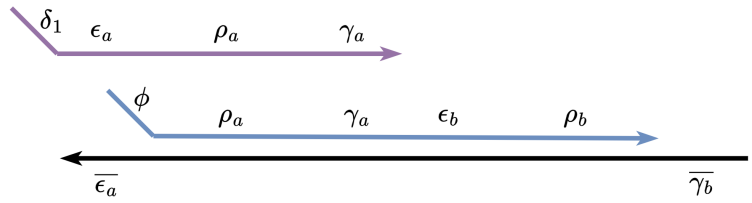
Step 2b: System after Washing



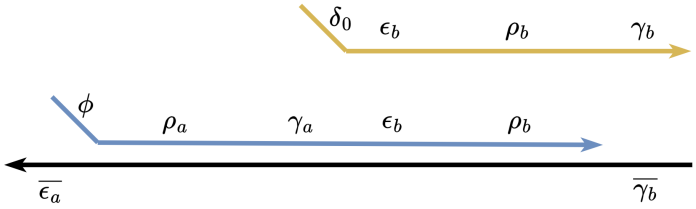
Step 3: Introduce retrieval strands with target delta



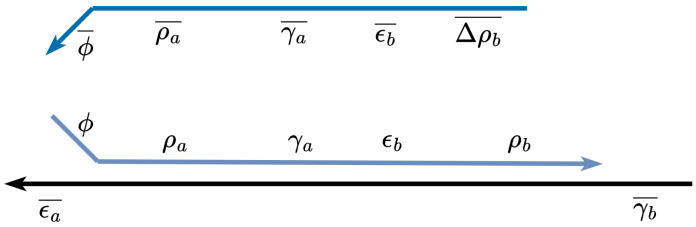
Step 4: Introduce replacement strand



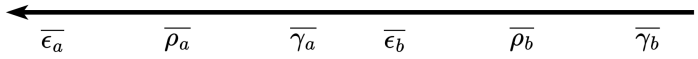
Step 5: Introduce other replacement strand



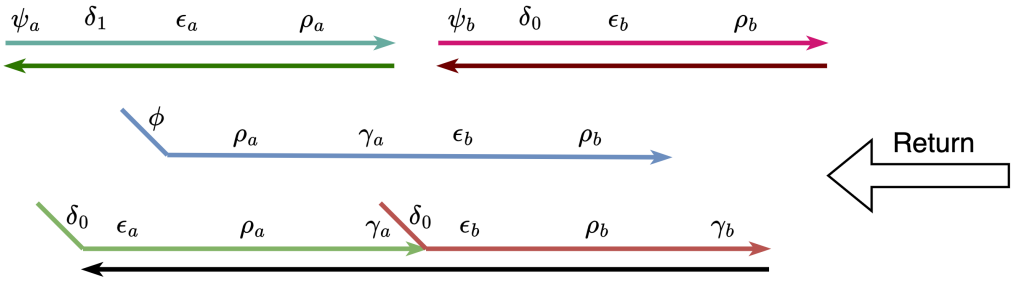
Step 6a: Retrieve the cover strand



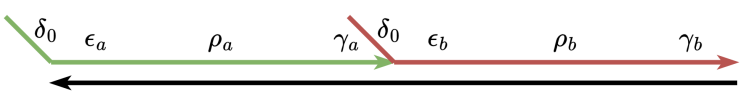
Step 6b: System after Wash



Step 7: Return second tube to main system



After Final Wash



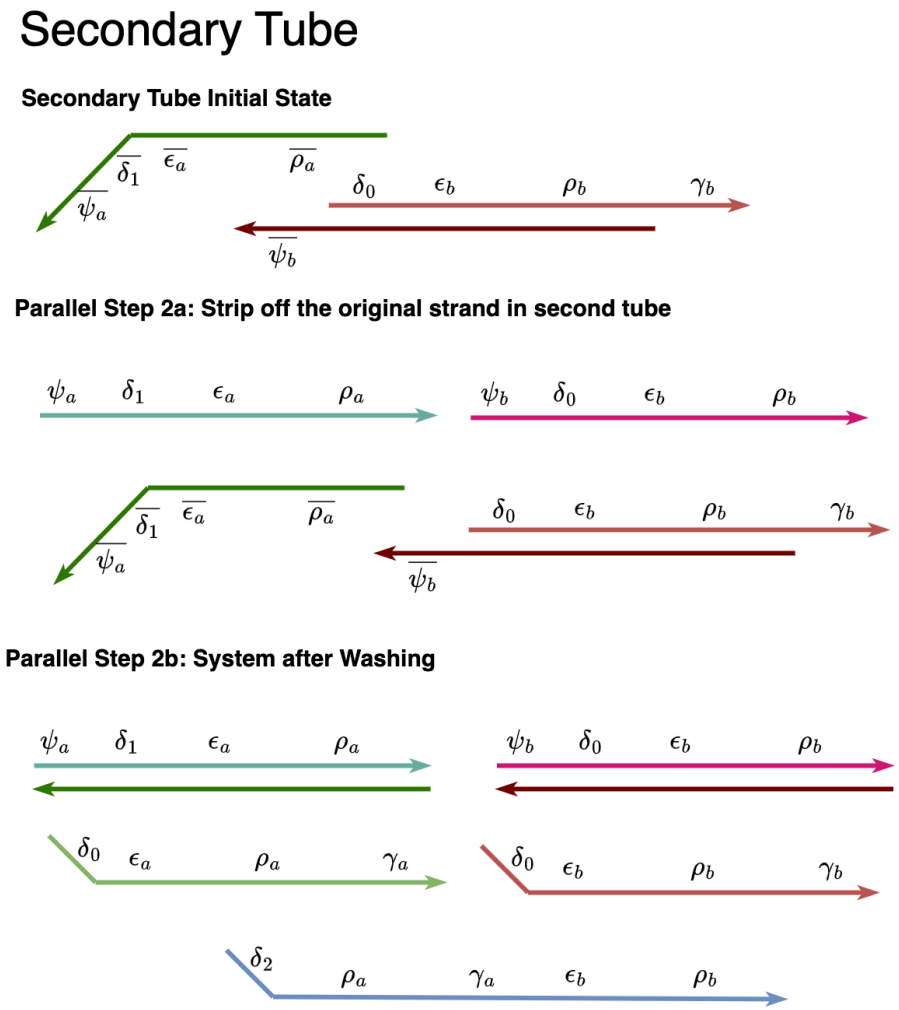


FIGURE A.10: An additional tube is required to preserve state within the system. The cover strand is introduced to the secondary tube during an isolation step.

Appendix B

Simulation Details

B.1 Simulation Options

Simulation were run at 25° Celsius and 10nM concentration unless specified otherwise. The `ordered-complex` stop condition was used for each kinetic simulation except the cooperative hybridization. The `count-by-domain` condition was required for cooperative hybridization reactions since the intermediate state were indistinguishable from reactants at the strand level. Since Multistrand is not compatible with coaxial stacking, just coaxial dangles, the `all_nupack3` historical ensemble mode was used to enable coaxial dangles[20]. The error threshold for rate predictions was set to 5% for all reactions. The dna04 parameters set was used for energy calculations.

B.2 Designing Orthogonal Systems in Nupack

Each *reg* in the encoding system can be designed as an orthogonal subsystem. `TargetTubes` were created for each instruction, as outlined in Sec 3.3, with an additional `cross-talk` tube to ensure orthogonality.

For each *reg* design, explicitly preventing interaction between δ_1 and $\overline{\delta_0}$ domains can simplify the design process. Including a `TargetTube` dedicated to the δ toehold with additional weighting improved design speed.

Diversity constraints were added to prevent the system from creating repetitive sections that would be difficult to synthesize.

1. **Diversity Constraint:** 2 different bases are required over a 5 nucleotide window.

B.3 Updating Multistrand and Migration

Prior to this project, Multistrand had not been updated since 2018, posing compatibility challenges with Nupack 4. To address this, initial evaluations were conducted by comparing Multistrand's equilibrium simulation results against those from Nupack, revealing discrepancies due to its continued reliance on Python 2 and outdated DNA parameterization models.

Multistrand was migrated to Python3, alongside a comprehensive test suite to ensure reliability. Additionally, the DNA parameter models within Multistrand were updated to align with the DNA 2004 parameters set[20].

B.4 Cover Strand Domains

In all simulations, the ϵ_b domain of the cover strand features a mismatch at the first base. This design choice is intended to enhance stability, allowing the cooperative hybridization reaction to overcome the fraying cause by the δ domain of reg_b. The mismatch creates a "bubble" that allows reg_b to stabilize, enabling the reg_a to complete the displacement more effectively.

However, this configuration posed challenges during pairwise computation attempts. Specifically, the cover strand struggled to displace the reg_a strand effectively, as evidenced by the `a_incumbent` operation exhibiting the lowest completion rate. This error may also be due to the coaxial stack bonus not being accounted for in the simulation model.

B.5 Identifying Leak States

We identify two primary leak pathways—sequential and dissociative—as detailed in Sec 2.7.

For each instruction operation, the strand introduced into the solution is categorized as the invading strand (strand C). We focus on the process where the incumbent strand (strand A) dissociates from the substrate (strand B). The incumbent strand binds to the invading strand

via the dissociative pathway. This interaction is further verified using Nupack, where the incumbent strand is excluded from binding with the substrate.

Additionally, we consider the intermediate complexes that form along the sequential dissociative pathway leading to the leak complex. By simulating the kinetics from the leak state back to the on-target complex, we estimate the leak reaction rate under detailed balance assumptions[14, 18]. The MFE state of each instruction is biased towards the on-target complex, suggesting a feasible pathway from the leak back to the on-target. This is confirmed through Multistrand kinetic simulations.

After determining the rate constants k_1 and k_2 for the transition from the leak back to the on-target complex, we use the equilibrium concentrations of the leak, intermediate, and on-target complexes to estimate the system's leak rate.

$$K_1 = \frac{[ABC]}{[AC] \cdot [B]} \tag{B.1}$$

$$K_2 = \frac{[A] \cdot [BC]}{[ABC]} \tag{B.2}$$

$$k_{-1} = \frac{k_1}{K_1} \tag{B.3}$$

$$k_{-2} = \frac{k_2}{K_2} \tag{B.4}$$

Appendix C

Applications

C.1 Reading Data

After isolating target data strands as described in Section 3.3.1, the stored values would be decoded using NGS technologies. This process is particularly effective with the Oxford Nanopore sequencing technique, which is renowned for its high throughput capabilities[16]. This technique supports parallel processing of multiple strands, significantly enhancing the efficiency of data reading.

C.2 Multi-Register Analysis

To assess the scalability of the design, a 4-bit system was developed for both storage and computation. The on-target concentration for each tube was averaged across the 4 registers, providing a comprehensive view of system performance.

C.2.1 Storage

The storage simulations demonstrated the system's potential as a room-temperature storage device, achieving an error rate of under 2% for each instruction on average. This low error rate suggests a robust system that could be effectively scaled up for enhanced storage capabilities.

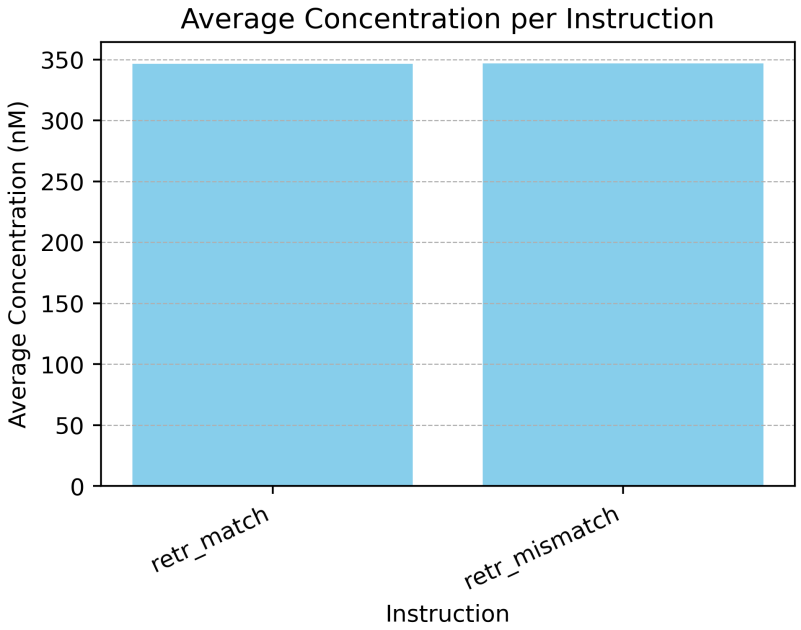


FIGURE C.1: Instruction averages across 4 registers with an initial concentration of 350nM, highlighting the on-target species concentration.

C.2.2 Compute

The computed equilibrium concentrations indicate that each instruction functions within the expected parameters, with errors kept under 5%. Similar to the 2-bit systems, instructions involving cooperative hybridization exhibited the most significant challenges in terms of completion rates. Despite this, all other instructions performed as anticipated, showcasing the system's capability to handle more complex operations involving multiple registers.

C.3 Sequence Possibilities

To further explore the scalability of the design, the Seqwalk tool was utilized to generate an orthogonal library. In this library, no substring of length k can be repeated, ensuring orthogonality across the sequences.

The sequence library generated cannot be directly utilized without further analysis. Nupack is required to verify that each sequence does not form undesirable secondary structures

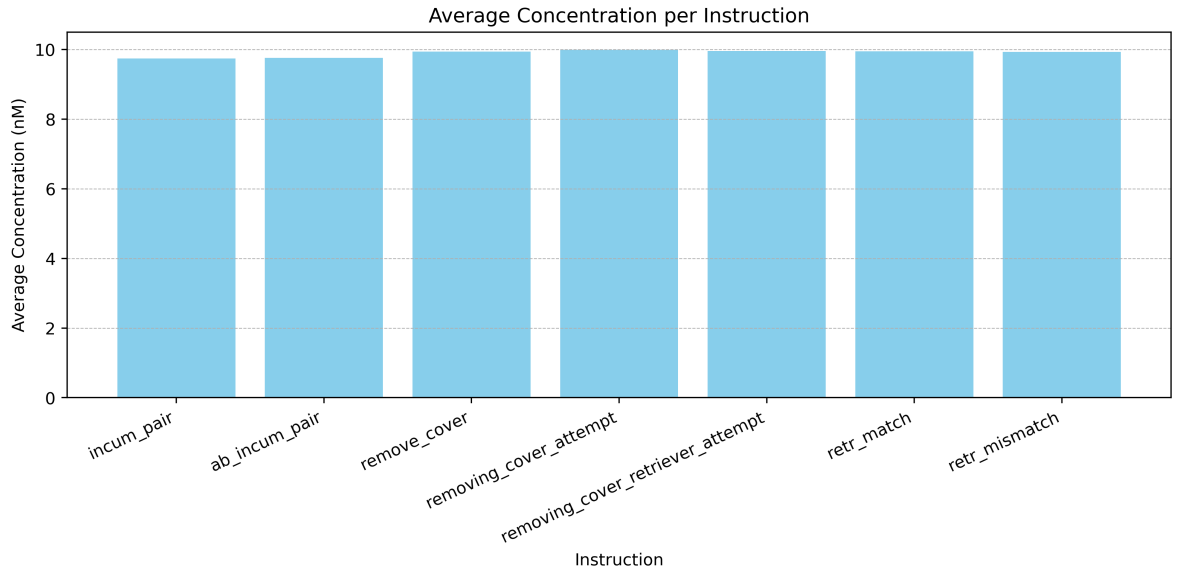


FIGURE C.2: Instruction averages across 4 registers with an initial concentration of 10nM, detailing the on-target species concentration.

within itself. These sequences could potentially be used for the ρ domains, as toehold require further design considerations.

C.4 Advanced Possibilities

Advanced compute operations may be devised using additional tubes for computation. Designing shorter sequences of backbones for *regs* not adjacent with each other could allow pair computation across the entire system. This opens the possibility of scaling the system by breaking the backbone across tubes, such that the instructions can reach completion in shorter periods of time.

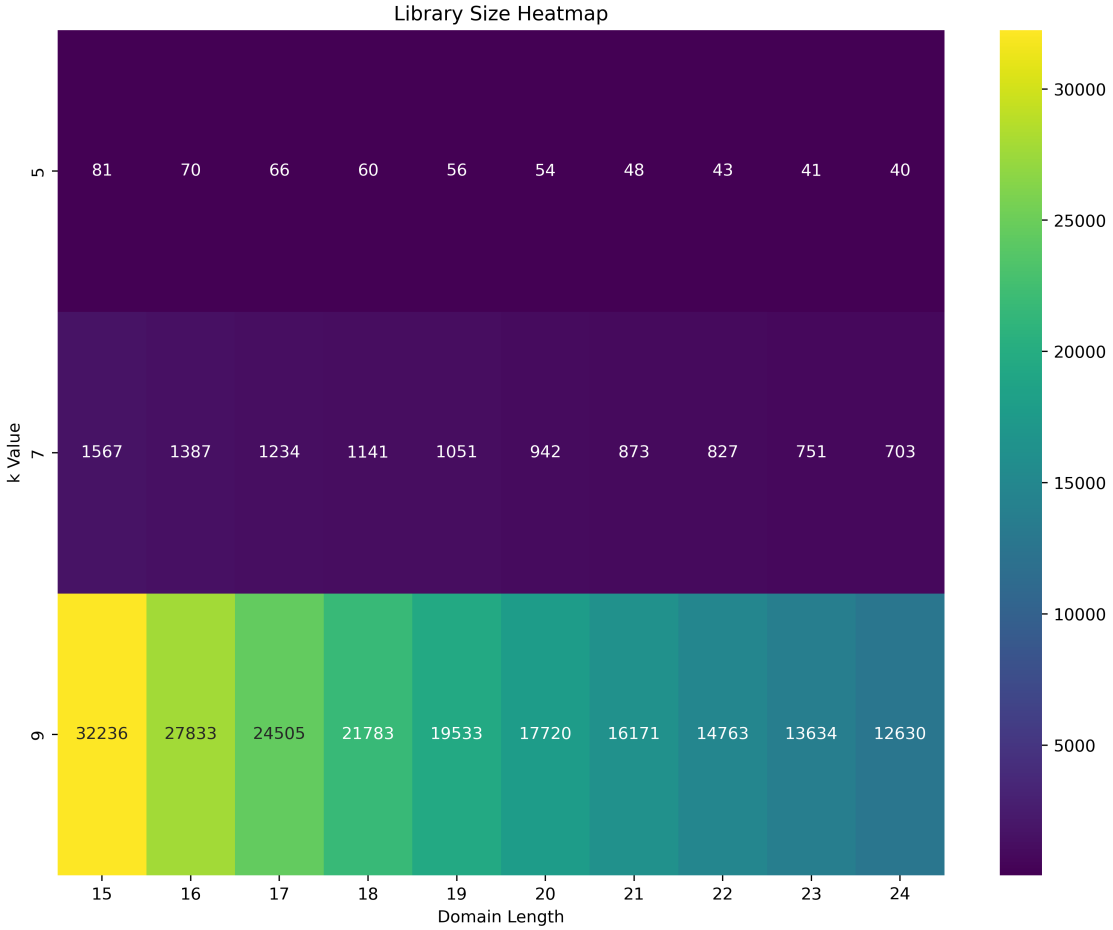


FIGURE C.3: The *k* SSM (Simple Shared Motifs) poses the tighter constraint in generating domain libraries.

Appendix D

Nupack Designed Sequences

Strand	Sequence
opt25.single_backbone	CTTAATATGAACTTGAAACCCTGG
opt25.reg	ATTTCAGCTCCCAGGGTTTCAAGTTCATATTAAG
opt25.retriever_match	ATGAACTTGAAACCCTGGGAGCTGAAAT
opt25.retriever_mismatch	ATGAACTTGAAACCCTGGAACTAACAAC
opt37.single_backbone	GGAATAATGAACTTGAAATGGGCT
opt37.reg	CCTCACACACAGCCCATTTCAAGTTCATTATTCC
opt37.retriever_match	ATGAACTTGAAATGGGCTGTGTGTGAGG
opt37.retriever_mismatch	ATGAACTTGAAATGGGCTCTCTAATCTG
opt11_7.single_backbone	ACTTAATATGAACTTGAAACCCTGGC
opt11_7.reg	CATTTCAGCTCGCCAGGGTTTCAAGTTCATATTAAGT
opt11_7.retriever_match	ATGAACTTGAAACCCTGGCGAGCTGAAATG
opt11_7.retriever_mismatch	ATGAACTTGAAACCCTGGCAACTAACAACC

TABLE D.1: Single Register Storage Sequence

Strand	Sequence
single_a_backbone	GATATACAGAGACAAATTAAACAG
reg_a	GGTTTGGTTTCTGTTTAATTTGTCTCTGTATATC
retriever_a_match	CAGAGACAAATTAAACAGAAACCAAACC
retriever_a_mismatch	CAGAGACAAATTAAACAGGAAAGGAAAG
single_b_backbone	CTTAATCTTCACATCACACACACC
reg_b	AGAGAAGAGAGGTGTGTGTGATGTGAAGATTAAG
retriever_b_match	CTTCACATCACACACACCTCTCTTCTCT
retriever_b_mismatch	CTTCACATCACACACACCCAACAACCCA
single_c_backbone	CTTAATATGAACTTGAAACCCTGG
reg_c	ATTTCAGCTCCCAGGGTTTCAAGTTCATATTAAG
retriever_c_match	ATGAACTTGAAACCCTGGGAGCTGAAAT
retriever_c_mismatch	ATGAACTTGAAACCCTGGAACTAACAAC
single_d_backbone	GTAAATCAATAAGGTAAAGGAAAG
reg_d	TTTCGTCTTTCTTTCCTTTACCTTATTGATTTAC
retriever_d_match	CAATAAGGTAAAGGAAAGAAAGACGAAA
retriever_d_mismatch	CAATAAGGTAAAGGAAAGGCACAATAAA

TABLE D.2: Multi-Register Storage Sequences

Strand	Sequence
single_a_backbone	CATTTACCTTTACCTCCATT
reg_a	GGTAAGAGGGAATGGAGGTAAAGGTAAATG
retriever_a_match	CCTTTACCTCCATTCCCTCTTACC
retriever_a_mismatch	CCTTTACCTCCATTTCAACTTCAA
single_b_backbone	GTTGTAGTTTGAGTTATGAG
reg_b	CAATCCACTACTCATAACTCAAACTACAAC
retriever_b_match	GTTTGAGTTATGAGTAGTGGATTG
retriever_b_mismatch	GTTTGAGTTATGAGGGCGCGTTTA
pair_a_b_backbone	GTTGTAGTTTGAGTTATGAGCATTTACCTTTACCTCCATT

TABLE D.3: Pair Computational Sequences

Strand	Sequence
single_a_backbone	GAATAAGACAAGAACAAGAACACG
reg_a	CCTTTATTTGCGTGTTCTTGTTCTTGTCTTATTC
retriever_a_match	GACAAGAACAAGAACACGCAAATAAAGG
retriever_a_mismatch	GACAAGAACAAGAACACGGGAAGATAAA
single_b_backbone	GTTTACCTGATTGACGACTGCTAG
reg_b	GGGAAATAGGCTAGCAGTCGTCAATCAGGTAAAC
retriever_b_match	CTGATTGACGACTGCTAGCCTATTTCCC
retriever_b_mismatch	CTGATTGACGACTGCTAGATTATATTAC
cover_a_b	CTTCTCCTCCCTTGTTCTTGTCTTATTCCTAGCGGTCGTCAATCAG
pair_a_b_backbone	GTTTACCTGATTGACGACTGCTAGGAATAAGACAAGAACAAGAACACG
cover_retriever_a_b	GACGACCGCTAGGAATAAGACAAGAACAAGGGAGGAGAAG
single_c_backbone	GAAAGTAAAGGTGGAGGTGTAAAG
reg_c	CTCCCATTCACTTTACACCTCCACCTTTACTTTC
retriever_c_match	AAAGGTGGAGGTGTAAAGTGAATGGGAG
retriever_c_mismatch	AAAGGTGGAGGTGTAAAGAGAGTAAAGA
cover_b_c	CCATCCCTTCGTCGTCAATCAGGTAAACGTTTACACCTCCACCTTT
pair_b_c_backbone	GAAAGTAAAGGTGGAGGTGTAAAGGTTTACCTGATTGACGACTGCTAG
cover_retriever_b_c	GGAGGTGTAAACGTTTACCTGATTGACGACGAAGGGATGG
single_d_backbone	GTTGTAGGTAGAGGTGGAGTGGAG
reg_d	CCCTTAACCACTCCACTCCACCTCTACCTACAAC
retriever_d_match	GGTAGAGGTGGAGTGGAGTGGTTAAGGG
retriever_d_mismatch	GGTAGAGGTGGAGTGGAGAAGAAAGAAG
cover_c_d	CCAATATCCAACCTCCACCTTTACTTTCCGCCACTCCACCTCTACC
pair_c_d_backbone	GTTGTAGGTAGAGGTGGAGTGGAGGAAAGTAAAGGTGGAGGTGTAAAG
cover_retriever_c_d	GGTGGAGTGGCGGAAAGTAAAGGTGGAGGTTGGATATTGG

TABLE D.4: Multi-Register Computation Sequences

Bibliography

[1] Stefan Badelt et al. "A domain-level DNA strand displacement reaction enumerator allowing arbitrary non-pseudoknotted secondary structures". In: *Journal of The Royal Society Interface* 17.167 (2020), p. 20190866. DOI: 10.1098/rsif.2019.0866. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsif.2019.0866. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2019.0866.

[2] Joseph Berleant et al. "Automated sequence-level analysis of kinetics and thermodynamics for domain-level DNA strand-displacement systems". In: *Journal of The Royal Society Interface* 15.149 (2018), p. 20180107. DOI: 10.1098/rsif.2018.0107. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsif.2018.0107. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2018.0107.

[3] UC-Davis Molecular Computing Group. *gpac*.

[4] Sean Kerwin. "Nucleic Acids: Structures, Properties, and Functions". In: *Journal of Medicinal Chemistry - J MED CHEM* 43 (Nov. 2000), pp. 4721–4722. DOI: 10.1021/jm000396p.

[5] Herr AJ. Preston BD Albertson TM. "DNA Replication Fidelity and Cancer". In: *Semin Cancer Biol.* 2010.

[6] Lulu Qian and Erik Winfree. "Scaling Up Digital Circuit Computation with DNA Strand Displacement Cascades". In: *Science* 332.6034 (2011), pp. 1196–1201. DOI: 10.1126/science.1200520. eprint: https://www.science.org/doi/pdf/10.1126/science.1200520. URL: https://www.science.org/doi/abs/10.1126/science.1200520.

[7] L P Reynaldo et al. "The kinetics of oligonucleotide replacements." eng. In: *J Mol Biol* 297.2 (24), pp. 511–520. ISSN: 0022-2836 (Print); 0022-2836 (Linking). DOI: 10.1006/jmbi.2000.3573.

[8] SAM ROWEIS, ERIK WINFREE, and RICHARD BURGOYNE. "A Sticker-Based Model for DNA Computation". In: *Journal of Computational Biology* 5 (1998).

[9] John Jr SantaLucia and Donald Hicks. "The thermodynamics of DNA structural motifs." eng. In: *Annu Rev Biophys Biomol Struct* 33 (2004), pp. 415–440. ISSN: 1056-8700 (Print); 1056-8700 (Linking). DOI: 10.1146/annurev.biophys.32.110601.141800.

[10] Joseph Malcolm Schaeffer, Chris Thachuk, and Erik Winfree. "Stochastic Simulation of the Kinetics of Multiple Interacting Nucleic Acid Strands". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2015, pp. 194–211. DOI: 10.1007/978-3-319-21999-8_13. URL: https://doi.org/10.1007%2F978-3-319-21999-8_13.

[11] John Shalf. "The future of computing beyond Moore's Law". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 378.2166 (2020), p. 20190061. DOI: 10.1098/rsta.2019.0061. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2019.0061. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2019.0061.

[12] Niranjan Srinivas et al. "On the biophysics and kinetics of toehold-mediated DNA strand displacement." eng. In: *Nucleic Acids Res* 41.22 (2013), pp. 10641–10658. ISSN: 1362-4962 (Electronic); 0305-1048 (Print); 0305-1048 (Linking). DOI: 10.1093/nar/gkt801.

[13] Boya Wang, Cameron Chalk, and David Soloveichik. "SIMD||DNA: Single Instruction, Multiple Data Computation with DNA Strand Displacement Cascades: Single Instruction, Multiple Data Computation with DNA Strand Displacement Cascades". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2019, pp. 219–235. DOI: 10.1007/978-3-030-26807-7_12. URL: https://doi.org/10.1007%2F978-3-030-26807-7_12.

[14] Boya Wang et al. "Effective design principles for leakless strand displacement systems". In: *Proceedings of the National Academy of Sciences* 115.52 (2018), E12182–E12191. DOI: 10.1073/pnas.1806859115. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1806859115. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1806859115.

[15] Boya Wang et al. "Parallel molecular computation on digital data stored in DNA". In: *Proceedings of the National Academy of Sciences* 120.37 (2023), e2217330120. DOI: 10.1073/pnas.2217330120. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.2217330120. URL: https://www.pnas.org/doi/abs/10.1073/pnas.2217330120.

[16] Yunhao Wang et al. "Nanopore sequencing technology, bioinformatics and applications". In: *Nature Biotechnology* 39.11 (2021), pp. 1348–1365. DOI: 10.1038/s41587-021-01108-x. URL: https://doi.org/10.1038/s41587-021-01108-x.

[17] J. D. WATSON and F. H. C. CRICK. "Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid". In: *Nature* 171.4356 (1953), pp. 737–738. DOI: 10.1038/171737a0. URL: https://doi.org/10.1038/171737a0.

[18] Jin Yang et al. "On imposing detailed balance in complex reaction mechanisms." eng. In: *Biophys J* 91.3 (2006), pp. 1136–1141. ISSN: 0006-3495 (Print); 1542-0086 (Electronic); 0006-3495 (Linking). DOI: 10.1529/biophysj.105.071852.

[19] Dina Zielinski Yaniv Erlich. "DNA Fountain enables a robust and efficient storage architecture". In: *Science*. Vol. 355. 2017.

[20] Joseph N Zadeh et al. "NUPACK: Analysis and design of nucleic acid systems." eng. In: *J Comput Chem* 32.1 (2011), pp. 170–173. ISSN: 1096-987X (Electronic); 0192-8651 (Linking). DOI: 10.1002/jcc.21596.

[21] David Yu Zhang. "Cooperative Hybridization of Oligonucleotides". In: *Journal of the American Chemical Society* 133.4 (Feb. 2011), pp. 1077–1086. DOI: 10.1021/ja109089q. URL: https://doi.org/10.1021/ja109089q.

[22] David Yu Zhang and Erik Winfree. "Control of DNA Strand Displacement Kinetics Using Toehold Exchange". In: *Journal of the American Chemical Society* 131.47 (Dec. 2009), pp. 17303–17314. DOI: 10.1021/ja906987s. URL: https://doi.org/10.1021/ja906987s.

[23] Sedigheh Zolaktaf et al. "Inferring Parameters for an Elementary Step Model of DNA Structure Kinetics with Locally Context-Dependent Arrhenius Rates". In: *DNA Computing and Molecular Programming*. Ed. by Robert Brijder and Lulu Qian. Cham: Springer International Publishing, 2017, pp. 172–187. ISBN: 978-3-319-66799-7.