Efficiently Estimating Kinetics of Interacting Nucleic Acid Strands Modeled as Continuous-Time Markov Chains

by

Sedigheh Zolaktaf

M.Sc, The University of British Columbia, 2015

B.Sc, Sharif University of Technology, 2013

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

The University of British Columbia

(Vancouver)

December 2020

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the thesis entitled:

Efficiently Estimating Kinetics of Interacting Nucleic Acid Strands Modeled as Continuous-Time Markov Chains

submitted by **Sedigheh Zolaktaf** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy** in **Computer Science**.

Examining Committee:

Anne Condon, Computer Science, University of British Columbia
Co-supervisor

Mark Schmidt, Computer Science, University of British Columbia
Co-supervisor

Gabriela V. Cohen Freue, Statistics, University of British Columbia
University Examiner

Steven Hallam, Microbiology, University of British Columbia
University Examiner

Petr Šulc, Molecular Sciences, Arizona State University
External Examiner

Additional Supervisory Committee Members:

Alexandre Bouchard-Côté, Statistics, University of British Columbia
Supervisory Committee Member

Abstract

Nucleic acid molecules are vital constituents of living beings. These molecules are also utilized for building autonomous nanoscale devices with biological and technological applications, such as toehold switches, algorithmic structures, robots, and logic gates. Predicting the kinetics (non-equilibrium dynamics) of interacting nucleic acid strands, such as hairpin opening and strand displacement reactions, would assist with understanding the functionality of nucleic acids in the cell and with building nucleic-acid based devices.

Continuous-time Markov chains (CTMC) are commonly used to predict the kinetics of these reactions. However, predicting kinetics with CTMC models is challenging. Because, first, the CTMCs should be defined with accurate and biophysically realistic kinetic models. Second, the state space of the CTMCs may be large, making predictions time-consuming, particularly for reactions that happen on a long time scale (rare events), such as strand displacement at room temperature.

We introduce an Arrhenius kinetic model of interacting nucleic acid strands that relates the activation energy of a state transition with the immediate local environment of the affected base pair. Our model can be used in stochastic simulations to estimate kinetic properties and is consistent with existing thermodynamic models that make equilibrium predictions. We infer the model's parameters on a wide range of reactions by using mean first passage time (MFPT) estimates. We estimate MFPTs using exact computations on simplified state spaces. We show that our new model surpasses the performance of the previously established Metropolis kinetic model.

We further address MFPT estimation and the rapid evaluation of perturbed

parameters for parameter inference in the full state space of reactions' CTMCs. We show how to use a reduced variance stochastic simulation algorithm (RVSSA) to estimate MFPTs. We also introduce a fixed path ensemble inference (FPEI) approach for the rapid evaluation of perturbed parameters. These methods are promising, but they are not suitable for rare events. Thus, we introduce the pathway elaboration method, a time-efficient and probabilistic truncated-based approach for addressing both mentioned tasks. We demonstrate the effectiveness of our methods by conducting computational experiments on nucleic acid kinetics measurements that cover a wide range of rates for different type of reactions.

Lay Summary

Nucleic acids are vital constituents of living beings. With the advantage of having programmable configurations and movement, nucleic acids are also utilized for building autonomous nanoscale devices. For example, RNA biosensors can detect viruses and DNA robots are promising tools for drug delivery in cancer therapy. Predicting the kinetics (non-equilibrium dynamics) of reactions involving interacting nucleic acid strands would assist with building such devices. In this direction, accurate kinetic models and efficient prediction methods are desirable. We introduce a kinetic model that surpasses the performance of a well-established kinetic model and we show how to calibrate the models on various type of reactions. Moreover, we introduce a time-efficient probabilistic method for predicting the kinetics of reactions in large state spaces. Our contributions will make it easier to design nucleic acids with novel forms of movement, which can ultimately lead to practical nanoscale devices.

Preface

This thesis is the result of three collaborative research papers of researchers from the University of British Columbia and California Institute of Technology. Chapters 1, 2, and 6 are based on Zolaktaf et al. (2017, 2019, 2020), Chapter 3 is based on Zolaktaf et al. (2017), Chapter 4 is based on Zolaktaf et al. (2019), and Chapter 5 is based on Zolaktaf et al. (2020). My contributions are as follows:

- Zolaktaf et al. (2017): S. Zolaktaf, F. Dannenberg, X. Rudelis, A. Condon, J. M. Schaeffer, M. Schmidt, C. Thachuk, and E. Winfree. Inferring parameters for an elementary step model of DNA structure kinetics with locally context-dependent Arrhenius rates. In DNA Computing and Molecular Programming, Lecture Notes in Computer Science, volume 10467, pages 172–187, 2017.

  - I am the primary contributor in formulating the solution, incorporating the data, conducting experimental studies, and writing the paper. The other co-authors contributed in guiding the experimental studies, formulating the solution, incorporating the data, and writing the paper. Xander Rudelis also contributed in preliminary experimental studies. The Arrhenius kinetic model is solely developed by the Winfree Group.

- Zolaktaf et al. (2019): S. Zolaktaf, F. Dannenberg, E. Winfree, A. Bouchard-Côté, M. Schmidt, and A. Condon. Efficient parameter estimation for DNA kinetics modeled as continuous-time Markov chains. In DNA Computing and Molecular Programming, Lecture Notes in Computer Science, volume 11648, pages 80–99, 2019.

  - I am the primary contributor in formulating the solution, incorporating the data, conducting experimental studies, and writing the paper. The other co-authors contributed in guiding the experimental studies, formulating the solution, incorporating the data, and writing the paper. Frits Dannenberg also contributed in preliminary experimental studies.

- Zolaktaf et al. (2020): S. Zolaktaf, F. Dannenberg, M. Schmidt, A. Condon, and E. Winfree. The pathway elaboration method for mean first passage time estimation in large continuous-time Markov chains with applications to nucleic acid kinetics. In submission, 2020.

  - I and Frits Dannenberg are equal primary contributors. I contributed in formulating the solution, incorporating the data, conducting experimental studies, and writing the paper. Frits Dannenberg contributed in formulating the solution, incorporating the data, conducting and guiding experimental studies, and writing the paper. The other co-authors contributed in guiding the experimental studies, formulating the solution, incorporating the data, and writing the paper.

Contents

List of Tables

List of Figures

Acknowledgments

I am eternally grateful to Anne Condon and Mark Schmidt, whom I have been fortunate to have as my supervisors. I am thankful beyond words for your inspiration, guidance, help, time, patience, and kind support throughout my graduate studies. Thank you for teaching me how to become a better researcher and a better person. I extend my sincere gratitude to Erik Winfree, whom I have been fortunate to collaborate with throughout this work. I am thankful for your inspiration, guidance, help, and all the wonderful meetings over the years. Thank you for giving me the opportunity to learn from you. I am grateful for Frits Dannenberg, whose contributions and guidance have made this thesis possible. It has been a pleasure working with you and learning from you. I am grateful to Alexandre Bouchard-Côté whom I have had the great pleasure to collaborate with and learn from and for agreeing to serve on my supervisory committee. I am also grateful to my other collaborators, Chris Thachuk, Joseph M. Schaeffer, and Xander Rudelis, for their contributions to this thesis.

I am grateful to Gabriela V. Cohen Freue and Steven Hallam for serving as the university examiners, to Petr Šulc for agreeing to be the external examiner of this thesis, and to Nicholas V. Swindale for chairing my defense. I am thankful for your time and helpful feedback to this thesis.

I would like to thank the Computer Science administrators and technical staff, especially Joyce Poon, Holly Kwon, and Trevor McLeod, for their assistance with my paper work and technical issues. I would also like to thank my lab-mates including Alireza, Amit, Chris, Mehrdad, Monir, Neil, Noushin, Greg, Hedayat, Hooman, Reza, Sharan, Shu, and Sikander, for creating a wonderful environment to work in.

Finally, I am grateful to my friends and family; my dear friend Maryam, my sisters Zeinab and Neda, my Amin, and my parents. Thank you for your being my source of inspiration, and for always motivating me, encouraging me, and supporting me to accomplish my dreams. I am blessed to have you.

Dedication

To my family.

Chapter 1

Introduction

Nucleic acids are vital constituents of living beings, and exist in the forms of deoxyribonucleic acid (DNA) and ribonucleic acid (RNA). DNA molecules contain the genetic information and instructions that enable the functioning and development of living organisms. RNA molecules are essential for decoding the genetic information contained within DNA. With the advantage of having programmable binding interactions and movement in vitro and in vivo, nucleic acids are also promising elements for building nanoscale biotechnological tools (Chen et al., 2015; Seeman and Sleiman, 2017; Simmel et al., 2019). For example, RNA toehold switches can activate gene expression (Green et al., 2014) and they can be used for the realization of paper-based biosensors to detect viruses (Pardee et al., 2016). As another example, DNA robots can diagnose and potentially treat cancerous activity using DNA origami (Li et al., 2018). Nucleic acids can also be used for building computing devices, such as digital circuits that compute the square root of a number (Qian and Winfree, 2011) and neural networks that recognize hand-written digits (Cherry and Qian, 2018).

Designers of these biotechnological and molecular programming tools must often pay careful attention to nucleic acid kinetics, in order to ensure that the tools work effectively. By kinetics, we mean non-equilibrium dynamics, such as the rate of a reaction, the order in which different strands interact, or secondary structures[1] that form when a system is not in thermodynamic equilibrium. Predicting

[1] A secondary structure refers to the hydrogen bonding state of the bases.

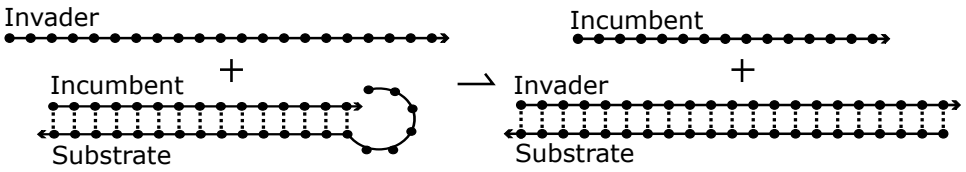


Figure 1.1: An example of a toehold-mediated three-way strand displacement reaction. An invader strand displaces an incumbent strand in a duplex through a strand displacement reaction.

the kinetics of reactions involving interacting nucleic acid strands would assist with building nucleic acid-based devices. For example, toehold-mediated three-way strand displacement reactions (Figure 1.1) are widely used for building nucleic acid-based devices (Simmel et al., 2019; Zhang and Seelig, 2011), such as digital circuits (Qian and Winfree, 2011), neural networks (Cherry and Qian, 2018; Qian et al., 2011), oscillators (Srinivas et al., 2017), and DNA robots (Thubagere et al., 2017). Other type of reactions that we use in this work (see Chapter 2) include helix association and dissociation, hairpin closing and opening, and four-way branch migration.

In this direction, developing accurate and efficient nucleic acid kinetic prediction tools is an important milestone (Angenent-Mari et al., 2020; Flamm et al., 2000; Ouldridge et al., 2011; Schaeffer et al., 2015; Šulc et al., 2012; Tang et al., 2005; Zhang et al., 2018). These tools are desirable for building nucleic acid-based devices whose nucleic acid sequences and experimental conditions need to be carefully designed to control their behaviour. They would allow many, though not all, unanticipated design flaws to be identified prior to conducting wet-lab experiments, and would allow more complex molecular devices to be designed and successfully implemented with fewer deficiencies needing to be debugged experimentally. However, predicting the kinetics of reactions is challenging and depends on the sequence of the strands and experimental conditions, making the design of artifacts challenging.

Because of these pressing needs, there has been great progress on kinetic simulators that can be used to simulate the movement of interacting nucleic acid strand in addition to predicting other kinetic properties, such as rate constants. The simu-

lators range from coarse-grained models that consider large rearrangements of the base pairs[2] (Tang et al., 2005) and often factor in tertiary structure, to elementary step models that consider the forming or breaking of a single base pair (Flamm et al., 2000; Schaeffer et al., 2015), to molecular dynamics models that follow the three-dimensional motion of the polymer chains (Ouldridge et al., 2011; Schreck et al., 2015; Šulc et al., 2012). Machine learning models, such as neural networks, have also been developed that predict kinetic properties (Angenent-Mari et al., 2020; Zhang et al., 2018). Both simulators and machine learning models need to be trained on experimental kinetics data. Elementary step models are of interest to us here because they are computationally more efficient than molecular dynamics, yet they also can represent and thus discover unexpected secondary structures that occur between any two possible secondary structures.

In modeling nucleic acid kinetics, continuous-time Markov chains (CTMCs) play a central role, such as the Kinfold (Flamm et al., 2000) and the Multistrand (Schaeffer, 2012; Schaeffer et al., 2015) kinetic simulators. A CTMC is a stochastic process on a discrete set of states that have the Markov property, so that future possible states are independent of past states given the current state. The time in a state before transitioning to another state (the holding time) is continuous; to retain the Markov property, holding times follow an exponential distribution with a single rate parameter for each state-to-state transition. CTMCs are widely used in natural and physical sciences, such as for modeling nucleic acid reactions (Dykeman, 2015; Flamm et al., 2000; Schaeffer et al., 2015), protein folding (McGibbon and Pande, 2015), chemical reaction networks (Anderson and Kurtz, 2011; Cappelletti et al., 2020; Soloveichik et al., 2008), and molecular evolution (Liò and Goldman, 1998). Particularly, in elementary step models of nucleic acid kinetics (Dykeman, 2015; Flamm et al., 2000; Schaeffer et al., 2015), states correspond to nucleic acid secondary structures and a transition between two states corresponds to the breaking or forming of a base pair. The transition rates are specified with kinetic models (Kawasaki, 1966; Metropolis et al., 1953) along with thermodynamic models that make equilibrium predictions (Andronescu et al., 2003; Bellaousov et al., 2013; Hofacker, 2003; Zadeh et al., 2011).

[2] A base pair refers to when two bases bound to each other by hydrogen bonds.

To make accurate and realistic kinetic predictions, accurate and biophysically realistic models of nucleic acid kinetics are desirable. Even though thermodynamic models (Andronescu et al., 2003; Hofacker, 2003; Xu and Mathews, 2016; Zadeh et al., 2011) have been extensively calibrated to experimental data (Andronescu et al., 2010; Mathews et al., 1999), extensive calibration of nucleic acid kinetic models remains challenging. Thus, one focus of this work is to develop a kinetic model that is more comprehensive than existing models and to show how to efficiently calibrate kinetic models on various type of reactions.

To evaluate and calibrate kinetic models, computational predictions need to be compared with experimental measurements. A central quantity of interest in CTMCs, which we use in this work, is the mean first passage time (MFPT) to reach a set of target states starting from a set of initial states. The MFPT is commonly used to estimate the rate of a process (Reimann et al., 1999; Schaeffer, 2013; Singhal et al., 2004). Predicting the rate of a process is advantageous, for example for designing devices that are controlled through rates of competing reactions. For a CTMC with a reasonable state space size, a matrix equation can provide an exact solution to the MFPT (Suhov and Kelbert, 2008). However, direct application of matrix methods is not feasible for estimating MFPTs of CTMCs that have large state spaces.

For such cases researchers may resort to stochastic simulation (Asmussen and Glynn, 2007; Doob, 1942; Gillespie, 1977, 2007; Ripley, 2009). For CTMCs in particular, the stochastic simulation algorithm (SSA) (Doob, 1942; Gillespie, 1977) is a widely used Monte Carlo procedure that can numerically generate statistically correct trajectories. In brief, the algorithm iteratively samples states based on transition rates and holding time of states. By sampling enough trajectories that reach a target state, an estimate of the MFPT can be obtained. In turn, direct simulation is inefficient for estimating MFPTs of reactions that happen on a long time scale (rare events), such as reactions that involve high-energy barrier states. A number of techniques have been developed for efficient sampling of simulation trajectories relevant to the event of interest (Allen et al., 2009; Bolhuis et al., 2002; Escobedo et al., 2009; Rubino and Tuffin, 2009; Shahabuddin, 1994), as will be discussed in Chapter 2. Such sampling techniques, unfortunately, must generally be re-run if model parameters change, which makes it costly to perform parameter scans or to

optimize a model.

CTMC systems may be treated by methods that truncate the state space to just a subset of "most relevant" states, so long as those states can be identified and they are few enough that matrix methods can compute the MFPT or other properties of interest (Kuntz et al., 2019; Munsky and Khammash, 2006). In such truncation-based methods, after the initial cost of enumerating states, the truncated CTMC can be reused to compute MFPTs for mildy perturbed parameters with accuracy relying on the truncated state space still containing the most relevant states. For example, we can reuse truncated CTMCs to speed up parameter inference or to optimize experimental conditions, such as the temperature, to obtain a desired functionality. The key challenge here is to efficiently enumerate a suitable subset of states that are sufficient for accurate estimation and also few enough that the matrix methods are tractable.

Thus, another focus of this work is to develop a method that efficiently addresses all three challenges for CTMCs: large state spaces, rare events, and efficient recomputation for perturbed model parameters.

1.1 Summary of Contributions

The contributions of this thesis are as follows:

- We curate experimentally determined reaction rate constants for nucleic acid kinetics from the literature, to conduct computational experiments in Chapters 3, 4, and 5. In Chapter 2, we describe the type of reactions that we use in this work. In Chapters 3, 4, and 5, we give an overview of the dataset that we curate for the experiments of the corresponding chapter.

- **Chapter 3:** We introduce an Arrhenius model of interacting nucleic acid kinetics that relates the activation energy of a state transition with the immediate local environment of the affected base pair. Our model can be used in stochastic simulations to estimate kinetic properties and is consistent with existing thermodynamic models. However, because explicit stochastic simulation can be extremely costly for obtaining sufficiently small error bars, we

employ reaction-specific simplified state spaces that enable MFPTs and reaction rate constants to be computed efficiently and exactly using matrix computations. The simplified state spaces are a strict subset of the full state space of Multistrand (Schaeffer, 2013; Schaeffer et al., 2015). We infer parameters for our model using an ensemble Markov chain Monte Carlo (MCMC) approach, in addition to a maximum a posteriori (MAP) approach, on a training dataset with 320 DNA kinetic measurements of hairpin closing and opening, helix association and dissociation, bubble closing and toehold-mediated strand exchange. Our new model surpasses the performance of the previously established Metropolis model both on the training set and on a testing set of size 56 composed of toehold-mediated three-way strand displacement with mismatches and hairpin opening and closing rates: reaction rates are predicted to within a factor of three for 93.4% and 78.5% of reactions for the training and testing sets, respectively.

Our framework and the dataset, as well as an online appendix that has additional experimental plots and analysis, are available at https://github.com/DNA-and-Natural-Algorithms-Group/ArrheniusInference.

- **Chapter 4:** We address MFPT estimation and the rapid evaluation of perturbed parameters for parameter inference in the full state space of reactions' CTMCs. We show how to use a reduced variance stochastic simulation algorithm (RVSSA), which is adapted from SSA, to estimate the MFPT of a reaction's CTMC. To speed up parameter estimation and the rapid evaluation of perturbed parameters, we introduce a fixed path ensemble inference (FPEI) approach, that we adapt from RVSSA. We show how to estimate model parameters in the full state of the reactions CTMCs using a generalized method of moments (GMM) estimator (Hansen, 1982). We conduct computational experiments on a dataset of 21 experimental DNA reactions that have moderate or large state spaces or are slow. In our experiments, FPEI speeds up parameter estimation compared to inference using SSA, by more than a factor of three for slow reactions. Also, for reactions with large state spaces, it speeds up parameter estimation by more than a factor of two.

  We implement RVSSA and FPEI using the Multistrand kinetic simulator

(Schaeffer, 2012; Schaeffer et al., 2015). Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/FPEI.

- **Chapter 5:** Similar to the previous chapter, we address MFPT estimation and the rapid evaluation of perturbed parameters in the full state space of reactions' CTMCs. However, we propose a method that is applicable to reactions that happen on a long time scale, that is rare events; Our method, called pathway elaboration, is a time-efficient probabilistic truncation-based approach for detailed-balance CTMCs. We demonstrate that pathway elaboration is suitable for predicting nucleic acid kinetics, by conducting computational experiments on 267 measurements that cover a wide range of rates for different types of DNA reactions, such as toehold-mediated three-way strand displacement and helix association. These measurements include reactions that have more than 100 bases in their strands. We utilize pathway elaboration to gain insight on the kinetics of two contrasting three-way strand displacement reactions. We compare the performance of pathway elaboration with the stochastic simulation algorithm (SSA) for MFPT estimation on 237 of the reactions for which SSA is feasible. We further use pathway elaboration to rapidly evaluate perturbed model parameters during optimization with respect to experimentally measured rates for these 237 reactions. The testing error on the remaining 30 reactions, which involved rare events and were not feasible to simulate with SSA, improved comparably with the training error.

  We implement pathway elaboration on top of the Multistrand kinetic simulator (Schaeffer, 2012; Schaeffer et al., 2015). Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/PathwayElaboration.

In this thesis, we only evaluate our work in the context of DNA reaction kinetics. However, we believe our Arrhenius kinetic model would also apply to RNA reaction kinetics. Moreover, even though we only evaluate the FPEI and pathway elaboration methods to DNA kinetics, they are generally applicable to other applications that are modeled as CTMCs, such as chemical reaction networks (Anderson and Kurtz, 2011; Cappelletti et al., 2020; Soloveichik et al., 2008), protein

folding (McGibbon and Pande, 2015), and molecular evolution (Liò and Goldman, 1998).

1.2 Thesis Outline

The rest of this thesis is organized as follows. In Chapter 2, we describe the preliminaries and the related work for this thesis. In Chapter 3, we propose the Arrhenius kinetic model and we give computational results. In Chapter 4, we propose the FPEI method and we give computational results. In Chapter 5, we propose the pathway elaboration method and we give computational results. Finally, in Chapter 6, we give a summary of our contributions.

Chapter 2

Preliminaries and Related Work

In this section, we first describe the most relevant concepts for continuous-time Markov chains (CTMCs). Then, we describe the Stochastic Simulation Algorithm (SSA) (Doob, 1942; Gillespie, 1977). After that, we describe the most relevant concepts for interacting nucleic acid strands, and we describe different type of nucleic acid reactions that we use in our dataset. Then, we explain how nucleic acid kinetics are modeled with CTMCs in the Multistrand kinetic simulator (Schaeffer, 2013; Schaeffer et al., 2015). Finally, we go over other related work.

2.1 Continuous-Time Markov Chain

Continuous-time Markov chain (CTMC). We define a CTMC as a tuple $\mathcal{C} = (\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}})$, where $\mathcal{S}$ is a countable set of states, $\mathbf{K} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$ is the rate matrix and $\mathbf{K}(s,s) = 0$ for $s \in \mathcal{S}$, $\pi_0 : \mathcal{S} \to [0,1]$ is the initial state distribution in which $\sum_{s \in \mathcal{S}} \pi_0(s) = 1$, and $\mathcal{S}_{\text{target}}$ is the set of target states. We define the set of initial states as $\mathcal{S}_{\text{init}} = \{s \in \mathcal{S} \mid \pi_0(s) \neq 0\}$. For CTMCs considered here, $\mathcal{S}_{\text{target}} \cap \mathcal{S}_{\text{init}} = \emptyset$. A transition between states $s, s' \in \mathcal{S}$ can occur only if $\mathbf{K}(s,s') > 0$. The probability of moving from state s to state s' is defined by the transition probability matrix $\mathbf{P} : \mathcal{S} \times \mathcal{S} \to [0,1]$ where

$$\mathbf{P}(s,s') = \frac{\mathbf{K}(s,s')}{\mathbf{E}(s,s)}. \tag{2.1}$$

Here $\mathbf{E} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$ is a diagonal matrix in which $\mathbf{E}(s,s) = \sum_{s' \in \mathcal{S}} \mathbf{K}(s,s')$ is the exit rate. The time spent in state s before a transition is triggered is then exponentially distributed with exit rate $\mathbf{E}(s,s)$. The generating matrix $\mathbf{Q} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is given by $\mathbf{Q} = \mathbf{K} - \mathbf{E}$.

Detailed-balance CTMC. In a detailed-balance CTMC $\mathcal{C}^R = (\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}}, \pi)$, also known as a reversible CTMC, a probability distribution $\pi : \mathcal{S} \to [0,1]$ over the states exists that satisfies the detailed balance condition

$$\pi(s)\mathbf{K}(s,s') = \pi(s')\mathbf{K}(s',s) \tag{2.2}$$

for all $s, s' \in \mathcal{S}$. The detailed balance condition is a sufficient condition for ensuring that π is a stationary distribution. A distribution π is called a stationary distribution if $\pi\mathbf{P} = \pi$. For a detailed-balance finite-state CTMC, π is the unique stationary distribution of the chain and is also the unique equilibrium distribution (Whitt, 2006).

Boltzmann distribution. In many Markov models of physical systems, eventually the population of states will stabilize and reach a Boltzmann distribution (Flamm et al., 2000; Schaeffer et al., 2015; Tang, 2010) at equilibrium. With this distribution, the probability that a system is in a state s is

$$\pi(s) = \frac{1}{Z} e^{-\frac{E(s)}{k_B T}}, \tag{2.3}$$

where $E(s)$ is the energy of the system at state s, T is the temperature, k_B is the Boltzmann constant, and $Z = \sum_{s \in \mathcal{S}} e^{-\frac{E(s)}{k_B T}}$ is the partition function.

To ensure that at equilibrium states are Boltzmann distributed, the detailed balance conditions are

$$\frac{\mathbf{K}(s,s')}{\mathbf{K}(s',s)} = e^{-\frac{E(s')-E(s)}{k_B T}}. \tag{2.4}$$

Reversible transition. In this work, a reversible transition between states s and s' means $\mathbf{K}(s,s') > 0$ if and only if $\mathbf{K}(s',s) > 0$, irrespective of the detailed balance condition.

Trajectories and paths. A trajectory (s_0,t_0), $(s_1,t_1,)$, ..., $(s_m,t_m,)$ with m transitions over a CTMC $\mathcal{C} = (\mathcal{S},\mathbf{K},\pi_0,\mathcal{S}_{\text{target}})$ is a sequence of states s_i and holding times t_i for which $\mathbf{K}(s_i,s_{i+1}) > 0$ and $t_i \in \mathbb{R}_{>0}$ for $i \geq 0$. We define a path $s_0,s_1,...,s_m$ with m transitions over a CTMC $\mathcal{C} = (\mathcal{S},\mathbf{K},\pi_0,\mathcal{S}_{\text{target}})$ as a sequence of states s_i for which $\mathbf{K}(s_i,s_{i+1}) > 0$.

Mean first passage time (MFPT). In a CTMC $\mathcal{C} = (\mathcal{S},\mathbf{K},\pi_0,\mathcal{S}_{\text{target}})$, for a state $s \in \mathcal{S}$ and a target state $s_{\text{f}} \in \mathcal{S}_{\text{target}}$, we define the MFPT τ_s to be the expected time to first reach s_{f} starting from state s. For each state s, the MFPT from s to s_{f} equals the expected holding time in state s plus the MFPT to s_{f} from the next visited state (Suhov and Kelbert, 2008), so

$$\tau_s = \frac{1}{\mathbf{E}(s,s)} + \sum_{s' \in \mathcal{S}} \frac{\mathbf{K}(s,s')}{\mathbf{E}(s,s)} \tau_{s'}. \tag{2.5}$$

Multiplying the equation by the exit rate $\mathbf{E}(s,s) = \sum_{s' \in \mathcal{S}} \mathbf{K}(s,s')$ then yields

$$\sum_{s' \in \mathcal{S}} \mathbf{K}(s,s')(\tau_{s'} - \tau_s) = -1. \tag{2.6}$$

Now writing $\mathbf{t} : \mathcal{S} \to \mathbb{R}_{\geq 0}$ to be the vector of MFPTs for each state, such that $\mathbf{t}[s] = \tau_s$, we find a matrix equation as

$$\tilde{\mathbf{Q}}\mathbf{t} = -\mathbf{1}, \tag{2.7}$$

where $\tilde{\mathbf{Q}}$ is obtained from $\mathbf{Q}$ by eliminating the row and column corresponding to the target state, and $\mathbf{1}$ is a vector of ones. If there exists a path from every state to the final state s_{f}, then $\tilde{\mathbf{Q}}$ is a weakly chained diagonally dominant matrix and is non-singular (Azimzadeh and Forsyth, 2016).

The MFPT from the initial states to the target state s_{f} is found from the initial distribution π_0 as

$$\tau_{\pi_0} = \sum_{s \in \mathcal{S}} \pi_0(s)\tau_s. \tag{2.8}$$

If instead of a single target state s_{f} we have a set of target states $\mathcal{S}_{\text{target}}$, then to compute the MFPT to $\mathcal{S}_{\text{target}}$ we convert all target states into one state s_{f} so that $\mathcal{S}^* = \mathcal{S} \setminus \mathcal{S}_{\text{target}} \cup \{s_{\mathrm{f}}\}$. For $s, s' \in \mathcal{S}^* \setminus \{s_{\mathrm{f}}\}$, we update the rate matrix $\mathbf{K}^* : \mathcal{S}^* \to \mathbb{R}_{\geq 0}$ by $\mathbf{K}^*(s, s_{\mathrm{f}}) = \sum_{s'' \in \mathcal{S}_{\text{target}}} \mathbf{K}(s, s'')$, $\mathbf{K}^*(s, s') = \mathbf{K}(s, s')$, and $\mathbf{K}^*(s_{\mathrm{f}}, s)$ is not used in the computation of the MFPT (see Eq. 2.7).

When the number of states of a CTMC is large, applying the matrix equations is not computationally feasible. Therefore, we use a subset of the states over the CTMC to build a *truncated CTMC*.

Truncated CTMC. Let $\hat{\mathcal{S}} \subseteq \mathcal{S}$ be a subset of the states over the CTMC $\mathcal{C} = (\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}})$ or detailed-balance CTMC $\mathcal{C}^R = (\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}}, \pi)$ and let $\hat{\mathcal{S}}_{\text{target}} \subseteq \hat{\mathcal{S}}$. We construct the rate matrix $\hat{\mathbf{K}} : \hat{\mathcal{S}} \times \hat{\mathcal{S}} \to \mathbb{R}_{\geq 0}$ as

$$\hat{\mathbf{K}}(s, s') = \mathbf{K}(s, s'). \tag{2.9}$$

We construct the initial probability distribution $\hat{\pi}_0 : \hat{\mathcal{S}} \to [0, 1]$ as

$$\hat{\pi}_0(s) = \frac{\pi_0(s)}{\sum_{s \in \hat{\mathcal{S}}} \pi_0(s)}. \tag{2.10}$$

We define the truncated CTMC as $\hat{\mathcal{C}} = (\hat{\mathcal{S}}, \hat{\mathbf{K}}, \hat{\pi}_0, \hat{\mathcal{S}}_{\text{target}})$ and $\hat{\mathcal{C}}^R = (\hat{\mathcal{S}}, \hat{\mathbf{K}}, \hat{\pi}_0, \hat{\mathcal{S}}_{\text{target}}, \hat{\pi})$ for $\mathcal{C}$ and $\mathcal{C}^R$, respectively. For a detailed-balance $\hat{\mathcal{C}}^R$, $\hat{\pi} : \hat{\mathcal{S}} \to [0, 1]$ defined as

$$\hat{\pi}(s) = \frac{\pi(s)}{\sum_{s \in \hat{\mathcal{S}}} \pi(s)}, \tag{2.11}$$

satisfies the detailed balance conditions in $\hat{\mathcal{C}}^R$ and is the unique equilibrium distribution of $\hat{\mathcal{S}}$ in $\hat{\mathcal{C}}^R$ (Whitt, 2006).

2.2 The Stochastic Simulation Algorithm

The stochastic simulation algorithm (SSA), which was developed by Doob (1942) and others and was popularized by Gillespie (1977) for simulating stochastic chem-

ical reactions, has been widely used to simulate statistically correct trajectories in CTMCs. The probability distribution of the states built from an infinite number of independent SSA simulations will be identical to the distribution of the states given by the master equation. SSA provides an unbiased and consistent estimate of the MFPT from an initial state to a target state. It estimates the MFPT as the mean of the first passage times of sampled trajectories. In brief, to sample a trajectory and its first passage time, SSA advances forward in two steps:

1. At a jump from the current state s_i, SSA samples the holding time T_i of the state from an exponential distribution with a rate equal to the sum of the transition rates from the state, in other words, $T_i \mid s_i \sim \text{Exp}(k_{s_i})$, where $k_{s_i} = \sum_{s \in \mathcal{S}} k_{s_i s}$, $\mathcal{S}$ is the state space of the CTMC, $k_{s_i s}$ is the transition rate from state s_i to state s, if s is not a neighbor of s_i then $k_{s_i s} = 0$, $\mathbb{E}[T_i \mid s_i] = k_{s_i}^{-1}$ and $\text{Var}(T_i \mid s_i) = k_{s_i}^{-2}$.

2. At a jump from the current state s_i, SSA samples the next state s_{i+1} from the outgoing transition probabilities of state s_i, in other words, $p(s_i, s) = \frac{k_{s_i s}}{k_{s_i}}, s_i \neq s$.

Let P be a trajectory of length Z from state s to state t, with holding times $T_1, ..., T_{Z-1}$, obtained by using SSA with initial state s, and ending the first time that state t is sampled. In SSA, the FPT of the trajectory is computed as

$$F^{\text{SSA}} = \sum_{i=1}^{Z-1} T_i. \tag{2.12}$$

By using N independently sampled trajectories, we obtain a Monte Carlo estimator for the MFPT of the CTMC as $\hat{\tau}_N^{\text{SSA}} = \frac{1}{N} \sum_{n=1}^{N} F_n^{\text{SSA}}$.

2.3 Interacting Nucleic Acid Strands

In this section, we first define interacting nucleic acid strands (reactions) based on the conventions of Multistrand (Schaeffer, 2013; Schaeffer et al., 2015) and then we describe the type of reactions that we use in this work.

We are interested in modeling the interactions of nucleic acid strands in a stochastic regime as in Multistrand (Schaeffer, 2013; Schaeffer et al., 2015). In this regime, we have a discrete number of nucleic acid strands (a set called Ψ^*) in a fixed volume V (the "box") and under fixed conditions, such as the temperature T and the concentration of Na^+ and Mg^{2+} cations. This regime can be found in systems that have a small volume with a fixed count of each molecule, and can also be applied to larger volumes when the system is well mixed. Moreover, it can be used to derive reaction rate constants of reactions in a chemical reaction network that follows mass-action kinetics (Schaeffer, 2013; Schaeffer et al., 2015).

A *complex* is a subset of strands of Ψ^* that are connected through base pairing. We refer to the complex base pairs, i.e., secondary structure, as the *complex microstate*. A *system microstate* is a set of complex microstates, such that each strand $\psi \in \Psi^*$ is part of exactly one complex. A *system macrostate* is a nonempty set of system microstates.

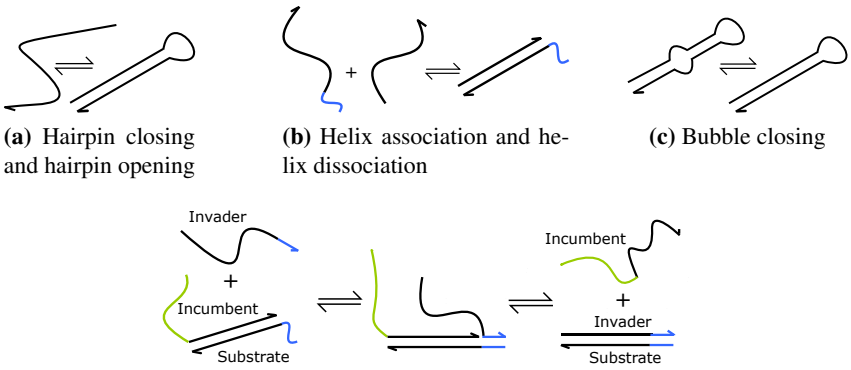
A *unimolecular reaction* with reaction rate constant k_1 has the form

$$A \xrightarrow{k_1} C + D, \tag{2.13}$$

and a *bimolecular reaction* with reaction rate constant k_2 has the form

$$B + F \xrightarrow{k_2} G + H. \tag{2.14}$$

Each reactant and product is a complex; A, B, C and G are nonempty but D and H may be empty complexes. For example, hairpin closing (Figure 2.1a) is a unimolecular reaction involving one strand, where complexes A and C are comprised of this one strand, while D is empty. Helix dissociation (Figure 2.1b) is an example of a unimolecular reaction where complex A has two strands while C and D are each of one of these strands. An example of a bimolecular reaction with two reactants and two non-empty products is three-way strand displacement reactions (Figure 2.1d). We are interested in computing k_1 and k_2 for such reactions. In Section 2.4, we describe how we can estimate them using MFPTs estimate from the Multistrand model.



(a) Hairpin closing and hairpin opening

(b) Helix association and helix dissociation

(c) Bubble closing

(d) Toehold-mediated three-way strand displacement

(e) Toehold-mediated four-way strand exchange

Figure 2.1: The type of reactions that we use in our dataset. Even though all reactions are reversible, in this work, we only consider the reverse reaction of hairpin closing, that is hairpin opening, and the reverse reaction of helix association, that is helix dissociation. In **(d)**, a possible intermediate state is shown where the toehold has fully bound. In **(e)**, two possible intermediate states are shown, where the toeholds have partially and fully bound.

2.3.1 Type of Reactions in Dataset

Here we introduce examples of interacting nucleic acid reactions that are of interest in biology and nanotechnology (Chen et al., 2015; Seeman and Sleiman, 2017; Simmel et al., 2019) and that we use in our experiments. Understanding the rates of reactions such as these has motivated our work.

Figure 2.1 shows an overview of the type of reactions that we consider in this work. The unimolecular reactions we consider are of the types hairpin closing and hairpin opening (Bonnet, 2000; Bonnet et al., 1998; Kim et al., 2006), helix dissociation (Cisse et al., 2012; Morrison and Stols, 1993; Reynaldo et al., 2000), and bubble closing (Altan-Bonnet et al., 2003). The bimolecular reactions we consider are of the types helix association (Hata et al., 2018; Morrison and Stols, 1993;

Zhang et al., 2018), toehold-mediated three-way strand displacement (Machinek et al., 2014; Reynaldo et al., 2000; Wetmur, 1976; Zhang and Winfree, 2009), and toehold-mediated four-way strand exchange (Dabby, 2013). These reactions are annotated with the temperature, the concentration of strands, and the concentration of Na^+ and Mg^{2+} cations in the buffer, which affect the reaction rate constants.

Hairpin closing and opening. In hairpin closing, a strand hybridizes itself and forms a hairpin loop. In hairpin opening, the reverse reaction of hairpin closing, the base pairs in the hairpin structure break to form a strand. Hairpin motifs are widely used in in biotechnology, such as in realizing molecular beacon probes (Tyagi and Kramer, 1996) and in realizing toehold switches (Green et al., 2014) and as fuels in autonomous nucleic-acid based devices (Green et al., 2006; Muscat et al., 2011; Venkataraman et al., 2007).

Helix association and dissociation. In helix association, two separate strands hybridize to form a duplex. In helix dissociation, the reverse reaction of helix association, two strands that have formed a duplex break base pairs to form two disconnected strands. Helix association and dissociation reactions are known to be fundamental to many cellular processes (Morrison and Stols, 1993) and are also commonly used in biotechnological applications (Khodakov et al., 2016; Lockhart et al., 1996).

Bubble closing. In this reaction, the bases of a bubble within a hybridized domain bond to form a fully hybridized domain. Bubbles are important examples of conformational change in nucleic acids (Hanke and Metzler, 2003).

Toehold-mediated three-way strand displacement. In this reaction, one of the strands in a duplex is replaced by an invader strand. The duplex consists of an incumbent strand and a complementary strand. In addition to the hybridized domain, the substrate strand also contains an unhybridized region called a *toehold* which facilitates the reaction; the invading strand usually binds to the toehold region of the substrate and then displaces the incumbent strand via three-way branch migration. The toehold can control the rate of the reaction by several orders of magnitude

by varying the sequence and the length of the toehold (Zhang and Winfree, 2009). Toehold-mediated three-way strand displacement reactions are widely used to build autonomous DNA devices, such as robots that sort molecular cargo (Thubagere et al., 2017), digital circuits (Qian and Winfree, 2011), oscillators (Srinivas et al., 2017), and neural networks (Cherry and Qian, 2018).

Toehold-mediated four-way strand exchange. In this reaction, two duplexes simultaneously exchange strands via four-way branch migration. The duplexes also have toehold domains that facilitate the reaction; the detached duplexes usually bind through the toehold region and then strand exchange between the duplexes occurs. The toeholds are known to control the reaction by several orders of magnitude (Dabby, 2013). Compared to toehold-mediated three-way strand displacement, toehold-mediated four-way strand exchange better prevents cross-talking between strands that are not supposed to interact (Dabby, 2013), which is desirable for building DNA devices. Toehold-mediated four-way strand exchange reactions have been used to implement autonomous locomotion (Venkataraman et al., 2007) and molecular probes (Duose et al., 2012).

The reactions may also have *mismatches*, in which the sequences of a duplex in an initial or a target complex are not perfectly complementary. The mismatches may effect the rate of a reaction by several orders of magnitude (Cisse et al., 2012; Machinek et al., 2014). Moreover, in some of the reactions, a sequence in a duplex may have a dangling end.

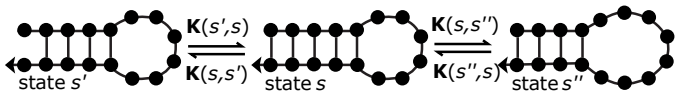


Figure 2.2: State s can transition to states s' and s'' by breaking a base pair. States s' and s'' can transition to state s by forming a base pair.

Rare reaction. A rare reaction is a reaction that happens on a long time scale, such as reactions in which initial states are separated from the final states by high-energy

barriers states. For example, the dissociation of a long duplex at room temperature could take a long time to complete.

2.4 The Multistrand Kinetic Simulator

Multistrand is a kinetic simulator (Schaeffer, 2013; Schaeffer et al., 2015) for analyzing the folding kinetics of multiple interacting nucleic acid strands. It can handle both a system of DNA strands and a system of RNA strands[1]. The Multistrand kinetic model is a detailed-balance CTMC $\mathcal{C}^R = (\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}}, \pi)$ for a set of interacting nucleic acid strands Ψ^* in a fixed volume V (the "box") and under fixed conditions, such as the temperature T and the concentration of Na^+ and Mg^{2+} cations. Currently, the state space $\mathcal{S}$ of the CTMC corresponds to the set of all non-pseudoknotted[2] system microstates of the set Ψ^* of interacting strands. Transitions between states correspond to elementary steps, that is the forming or breaking of a single base pair[3]. For example, in Figure 2.2, state s can transition to states s' and s'' by breaking a base pair. *Unimolecular transitions* are distinguished from *bimolecular transitions*. In a unimolecular transition, the number of strands in each complex remains constant. There are *bimolecular join moves*, where two complexes merge, and *bimolecular break moves*, where a complex falls apart and releases two separate complexes. The rate at which a transition triggers is determined by a kinetic model, such as the Metropolis kinetic model (Metropolis et al., 1953) (described in Section 2.4.1). The distribution π_0 is an initial distribution over the microstates of the reactant complexes, and the set $\mathcal{S}_{\text{target}}$ is a subset of the microstates of the product complexes, which we determine based on the type of the reaction.

Reaction rate constant estimation. Following the conventions of Multistrand

[1] Currently, Multistrand does not handle a system of mixed DNA and RNA strands, though it can be extended to handle such systems using good thermodynamic parameters.

[2] A pseudoknotted secondary structure has at least two base pairs in which one nucleotide of a base pair is intercalated between the two nucleotides of the other base pair. A non-pseudoknotted system microstates does not contain any pseudoknotted secondary structures. Currently, Multistrand excludes pseudoknotted secondary structures due to computationally difficult energy model calculations.

[3] Multistrand allows Watson-Crick base pairs to form, that is A-T and G-C in DNA and A-U and G-C in RNA. Additionally, it provides an option to allow G-T in DNA and G-U in RNA.

(Schaeffer, 2013), we estimate the reaction rate constant k_1 for a reaction in the form of Eq. 2.13 from its corresponding CTMC as

$$k_1 = \frac{1}{\tau_{\pi_0}}, \tag{2.15}$$

where we define τ_{π_0} as in in Eq. 2.8. We estimate the reaction rate constant for a reaction in the form of Eq. 2.14 from its corresponding CTMC as

$$k_2 = \frac{1}{u}\frac{1}{\tau_{\pi_0}}, \tag{2.16}$$

where u is the concentration of the reactants in the simulation. This equation is reasonable in the limit of low concentrations (Schaeffer, 2013).

2.4.1 The Metropolis Kinetic Model

The Metropolis model (Metropolis et al., 1953) is one of the kinetic rate models implemented in Multistrand (Schaeffer, 2012; Schaeffer et al., 2015). The Multistrand model considers a finite set of strands in a fixed volume (the "box") and defines the energy of a state as the sum of the standard free energy for each complex and a volume-dependent entropy term. For a state s containing N strands and M complexes, the free energy $\Delta G^\circ_{\text{box}}(s)$ is

$$\Delta G^\circ_{\text{box}}(s) = \sum_{c=1}^{M} \Delta G^\circ_{\text{complex}}(c) + (\mathcal{N} - \mathcal{M})\Delta G^\circ_{\text{volume}}, \tag{2.17}$$

where $\Delta G^\circ_{\text{complex}}(c)$ is the difference in Gibbs free energy[4] of complex c relative to the reference state and standard buffer conditions ($[\text{Na}^+] = 1$ M), and $\Delta G^\circ_{\text{volume}} = -RT \ln u$ is the loss of entropy resulting from fixing the position of a strand of concentration u relative to the standard concentration (1 M).

To ensure that simulations converge to the Boltzmann distribution over the states at equilibrium, the transition rates between any two adjacent states s and

[4]Calculated with thermodynamic models.

s' must satisfy detailed balance:

$$\mathbf{K}(s,s')/\mathbf{K}(s',s) = \exp\left\{-\left(\Delta G^{\circ}_{\text{box}}(s') - \Delta G^{\circ}_{\text{box}}(s)\right)/RT\right\}, \tag{2.18}$$

where $\mathbf{K}(s,s')$ is the transition rate from state s to state s', R is the gas constant, and T is the temperature.

In the Metropolis model, unimolecular transition rates are given by

$$\mathbf{K}(s,s') = \begin{cases} k_{\text{uni}} & \text{if } \Delta G^{\circ}_{\text{box}}(s') < \Delta G^{\circ}_{\text{box}}(s), \\ k_{\text{uni}} \exp\left(\frac{\Delta G^{\circ}_{\text{box}}(s) - \Delta G^{\circ}_{\text{box}}(s')}{RT}\right) & \text{otherwise,} \end{cases} \tag{2.19}$$

where $k_{\text{uni}} > 0$ is the unimolecular rate constant (units: s^{-1}). For bimolecular transitions $i \to j$ where two previously unconnected strands form a mutual base pair, the rate is given as

$$\mathbf{K}(s,s') = k_{\text{bi}} u, \tag{2.20}$$

and the rate of dissociation for the bimolecular transition $j \to i$ is given by

$$\mathbf{K}(s',s) = k_{\text{bi}} e^{-\frac{\Delta G^{\circ}_{\text{box}}(s) - \Delta G^{\circ}_{\text{box}}(s') + \Delta G^{\circ}_{\text{volume}}}{RT}} \times \text{M}, \tag{2.21}$$

where $k_{\text{bi}} > 0$ is the bimolecular rate constant (units: $\text{M}^{-1}\text{s}^{-1}$). $\theta = \{\ln k_{\text{uni}}, \ln k_{\text{bi}}\}$ are two free parameters in the model that are calibrated from experimental measurements (Morrison and Stols, 1993; Wetmur and Davidson, 1968). We emphasize that the rate of dissociation, Eq. 2.21, is independent of concentration u and $\Delta G^{\circ}_{\text{volume}}$, which follows from the definition of the free energy in a state (Eq. 2.17).

2.5 Other Related Methods

2.5.1 Kinetic Model Evaluation

Models of nucleic acid thermal stability have been extensively evaluated and calibrated to experimental data (Andronescu et al., 2010; Mathews et al., 1999) and

enable secondary structure software such as RNAsoft, ViennaRNA, RNAstructure, NUPACK, and mfold (Andronescu et al., 2003; Hofacker, 2003; Xu and Mathews, 2016; Zadeh et al., 2011; Zuker, 2003) to efficiently predict the equilibrium probabilities of nucleic acid secondary structures. By comparison, a similar calibration and evaluation of nucleic acid kinetic models to a broad range of measurements has not been attempted. Of particular interest is a study by Srinivas et al. (2013) which demonstrates that the Metropolis model of Multistrand is incompatible with observations of toehold-mediated strand displacement. Existing related work (Srinivas et al., 2013; Zhang and Winfree, 2009) successfully uses reaction specific models to calibrate a kinetic model. Unfortunately, reaction specific models are not easily adapted to other kinetic models or other type of reactions. We show how we can calibrate kinetic models on various type of reactions using rate estimates from the Multistrand model and using sampling and optimization methods.

2.5.2 Mean First Passage Time and Reaction Rate Constant Estimation

As explained in Section 2.1, we could compute the reaction rate constant of a reaction using the MFPT from the initial states to the target states in the reaction's CTMC. Alternatively, we could compute the reaction rate constant directly from measured data without modeling the reaction as a CTMC. For example, the weighted neighbour voting prediction algorithm has been developed to successfully predict hybridization rates (Zhang et al., 2018) from sequence, without enumerating the secondary structures of the reaction. We could also utilize neural networks, similar to Angenent-Mari et al. (2020) in which they successfully predict toehold switch function (Angenent-Mari et al., 2020). However, despite the accurate and fast computational prediction of these methods, to treat unseen type of reactions they would have to be adapted. On the other hand, the CTMC model of Multistrand can readily be applied to unimolecular and bimolecular reactions of interacting nucleic acid strands using a well-calibrated kinetic model. Also, Multistrand could provide unexpected intermediate states, which could be more than one, between any possible initial and target state. Using the neural networks models of Angenent-Mari et al. (2020), which use attention maps to interpret intermediate states, the number would be limited and the network would have to be adapted for

different initial and target states. Moreover, compared to neural network models, Multistrand generally has fewer free parameters[5], and thus would require fewer experimental data to be calibrated to.

As explained in Section 2.1, matrix equations can provide an exact solution to the MFPT of a CTMC that can be explicitly represented. However, it is infeasible to use matrix equations for CTMCs with large implicitly-represented state spaces. Alternatively, the MFPT could be obtained with SSA. However, SSA could be time consuming for events that happen on a long time scale, that is rare events. There exist numerous Monte Carlo techniques (Madras, 2002; Rubino and Tuffin, 2009) for driving simulations towards the target states or to reduce the variance of estimators. For example, importance sampling techniques (Andrieu et al., 2003; Doucet and Johansen, 2009; Hajiaghayi et al., 2014; Kuwahara and Mura, 2008; Shahabuddin, 1994) use an auxiliary sampler to bias the simulation, after which estimates are corrected with an importance weight. Moreover, many accelerated variants of SSA have been developed for CTMC models of chemically reacting systems (Cao et al., 2007; Gillespie, 2001, 2007), which can be adapted to simulate arbitrary CTMCs. There also exists a proliferation of rare event simulation methods for molecular dynamics (Allen et al., 2006, 2009; Bolhuis et al., 2002; Cabriolu et al., 2017; Elber, 2017; Weinan et al., 2002, 2005; Zuckerman and Chong, 2017). The ideas behind these methods can more or less be adapted for CTMCs and can be used along with SSA for more efficient computations. For example, the weighted ensemble (WE) (Huber and Kim, 1996; Zuckerman and Chong, 2017) approach has been used with SSA to estimate MFPTs (Donovan et al., 2013). In brief, in this approach, a number of non-overlapping bins are defined and a number of weighted trajectories are initiated. Within each bin the number of trajectories is held constant at a target number. At fixed time intervals, the trajectories are examined based on their location at that time. If a bin has fewer trajectories than the target number, then the trajectories are replicated so that the replicates carry an equal share of their parent's weight. Conversely, if a bin has more trajectories than the target number,

[5]For example, the Metropolis kinetic model has two kinetic parameters and the Arrhenius kinetic model that we introduce in Chapter 3 has 15 kinetic parameters. These kinetic models also depend on thermodynamic parameters which are obtained through thermodynamic prediction software, such as NUPACK (Zadeh et al., 2011), although they can also be treated as free parameters and be improved using Multistrand.

then the trajectories are pruned down to the target number and their weights are redistributed to the remaining trajectories. Stochastic simulations are usually not immediately reusable for the rapid evaluation of perturbed parameters and have to be adapted. In Chapter 4, we should how to adapt SSA for this purpose. In Chapter 5, we introduce the pathway elaboration method for successfully building truncated state spaces from initial state to targets states within a bounded running time.

2.5.3 Parameter Estimation and the Rapid Evaluation of Perturbed Parameters

In order to make estimations that are in accordance with measured data, the underlying models for the CTMCs should be calibrated to measured data. We could use simulation techniques that have been developed for efficient biased sampling (Allen et al., 2009; Bolhuis et al., 2002; Daigle et al., 2012; Escobedo et al., 2009; Shahabuddin, 1994). However, such sampling techniques would have to be generally re-run. Thus, we require a method which accurately and rapidly predicts the statistics of interest given a perturbation to the parameters. Any such method would also be useful for other tasks that require the rapid evaluation of perturbed parameters, such as designing a reaction to obtain a desired function.

Coarse-graining using Markovian state models have been effective (separately) for examining rare events (Sarich et al., 2014), but are mainly developed for molecular dynamics models. For example, Singhal et al. (2004) use transition path sampling (TPS) (Bolhuis et al., 2002) to build Markov state models for protein dynamics and to estimate MFPTs at perturbed temperatures using matrix equations (Singhal et al., 2004). In brief, in TPS an ensemble of paths are generated using a Monte Carlo procedure. First, a single path is generated that connects the initial and target states. New paths are then generated by picking random points along the current paths and running time-limited simulations from the points. Paths that reach either the initial or target state define possible new paths and the ones that do not are rejected. Even though we could use TPS along with SSA to simulate rare events for CTMCs (Eidelson and Peters, 2012), it is likely that many of the simulated paths could be rejected. For example, if the energy landscape has more than one local maxima between the initial and target states, then paths simulated from in between

these local maximums could require a very long simulation time to reach either the initial or the target states.

An important related quantity that has been widely studied is the transient probability of states, that is the probability distribution of the states over time. Transient probabilities are commonly used to calibrate CTMCs (Andrieu and Roberts, 2009; Loskot et al., 2019; Schnoerr et al., 2017) in nucleic acid kinetics (Hajiaghayi et al., 2014) and chemical reaction networks (Georgoulas et al., 2017; Golightly and Wilkinson, 2011; Horváth and Manini, 2008; Lück and Wolf, 2016). However, we use MFPT estimates since collecting a large dataset of MFPT estimates from the literature is feasible. Transient probabilities can be computed exactly with the master equation (Van Kampen, 1992) for CTMCs that have a feasible state space size. An important tool that has been developed to quantify the error of transient probability estimations for truncated CTMCs is the finite state projection (FSP) method (Munsky and Khammash, 2006). The FSP method tells us that as the size of the state space of the truncated CTMC grows, the approximation monotonically improves. Also, it guarantees that the approximate solution never exceeds the actual solution and provides bounds on the solution. As the authors of the FSP method mention, there are many ways to grow the state space, for example by iteratively adding states that are reachable from the state space within a fixed number of steps. However, applying matrix computations for very large state spaces could be time consuming. There have been many attempts to enumerate a suitable set of of states (Dinh and Sidje, 2016). In the Krylov-FSP-SSA approach (Sidje and Vo, 2015) an SSA approach is used to drive the FSP and adaptive Krylov methods are used to efficiently evaluate the matrix exponential. In brief, the method starts from an initial state space and proceeds iteratively in three steps. First, it drops low-probability states. Second, it runs SSA from each state of the remaining state space to incorporate probable states. Third, it adds states that are reachable within a fixed number of steps. Despite its great potential, this way of building the state space may not be suitable for rare events. However, our pathway elaboration method uses biased simulations to reach target states efficiently.

The idea of optimizing parameter sets by using truncated CTMCs has also been used with the Krylov-FSP-SSA method (Dinh and Sidje, 2017). Moreover, in related work (Georgoulas et al., 2017), an ensemble of truncated CTMCs is used to

obtain an unbiased estimator of transient probabilities, which are further used for Bayesian inference. Any success in building more efficient truncated CTMCs will also be useful in ensemble approaches.

Other types of truncation-based methods that are related to our work are probabilistic roadmap planning (Amato and Song, 2002; Kavraki et al., 1996; Tang, 2010; Tang et al., 2005) methods. These methods first sample a set of states according to some criteria, such as stability, to capture potentially important features. The states are then connected to nearby states to form a roadmap. To generate a truncated CTMC for MFPT estimation, one could enumerate all states that satisfy a certain criteria. For example, for nucleic acid reactions, one could enumerate all states below a certain free energy bound. However, this approach has two drawbacks. First, setting the boundary too low would mean the reaction pathway is not included in the state space, while setting the barrier too high could make the method inefficient as too many states are included. Second, this method would sample states irrespective of the transition rates. Instead, we rely on stochastic sampling from the initial states to the target states.

Chapter 3

The Arrhenius Kinetic Model

In this chapter, we report the initial results of our effort to develop accurate kinetic models for nucleic acids. We introduce the Arrhenius kinetic model.

3.1 Introduction

As explained in Chapters 1 and 2, models of nucleic acid thermal stability are calibrated to a wide range of experimental observations (Andronescu et al., 2010; Mathews et al., 1999), and typically predict equilibrium probabilities of nucleic acid secondary structures with reasonable accuracy (Andronescu et al., 2003; Hofacker, 2003; Xu and Mathews, 2016; Zadeh et al., 2011; Zuker, 2003). In comparison, a similar extensive calibration and evaluation of nucleic acid kinetic models has not been attempted so far, despite the development of kinetic models and simulation software such as Multistrand and Kinefold (Chen, 2008; Flamm et al., 2000; Schaeffer et al., 2015; Schreck et al., 2015; Xayaphoummine et al., 2005).

In this chapter, we develop a nucleic acid kinetic model based on Arrhenius dynamics that surpasses the performance of the well-established Metropolis kinetic model (Metropolis et al., 1953) (described in Chapter 2). It can be used in stochastic simulations and is consistent with existing thermodynamic models. A key difference of this model with the Metropolis model is the use of activation energy, which depends on the immediate local environment surrounding the affected base pair.

We conduct a preliminary study to assess whether the Arrhenius model is promising for predicting DNA kinetics, and to evaluate different calibration approaches. To calibrate and evaluate the Arrhenius and the Metropolis models, we compile a dataset of 376 experimentally determined reaction rate constants that we source from existing publications and cover a wide range of reactions, including hairpin closing, hairpin opening, bubble closing, helix association, helix dissociation, toehold-mediated three-way strand displacement, and toehold-mediated four-way strand exchange (Altan-Bonnet et al., 2003; Bonnet, 2000; Bonnet et al., 1998; Dabby, 2013; Kim et al., 2006; Machinek et al., 2014; Morrison and Stols, 1993; Reynaldo et al., 2000; Zhang and Winfree, 2009). To efficiently infer parameters and to obtain posterior parameter distributions, we use an ensemble Markov chain Monte Carlo (MCMC) approach. We also use a maximum a posteriori (MAP) approach. To evaluate the likelihood, we compare predicted reaction rate constants with experimental reaction rate constants. To estimate reaction rate constants from mean first passage times (MFPTs), we can use the stochastic simulation algorithm (SSA) (Doob, 1942; Gillespie, 1977) (described in Chapter 2). However, obtaining precise predictions using explicit stochastic simulation is computationally expensive, making MCMC parameter inference difficult. Instead, for each reaction, we employ a strict subset of its full state space in the Multistrand model (Schaeffer, 2013; Schaeffer et al., 2015), enabling MFPTs to be computed using matrix equations. Our simplified state spaces are based on 'zipper models' that were investigated previously to model DNA hybridization (Gibbs and DiMarzio, 1959). Overall, our results are encouraging and suggest that the new Arrhenius model is applicable to a wide range of DNA dynamic interactions and can be efficiently trained with our framework.

The rest of this chapter is organized as follows. Section 3.2 introduces our Arrhenius kinetic model, Section 3.3 introduces our kinetic dataset, Section 3.4 introduces our inference framework, Section 3.5 describes our results comparing the inferred parameters to the database of experimental measurements.

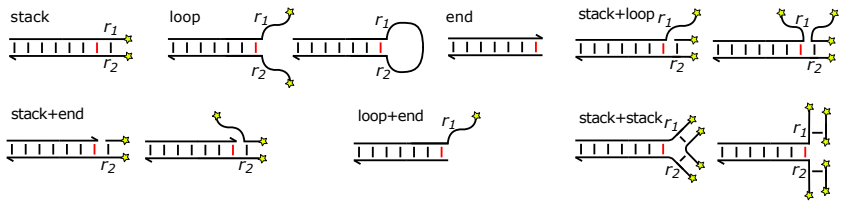


Figure 3.1: Seven type of local contexts that are model differentiates between. The right side of the red base pair forms one half of the local context. The classification of the half context depends on the pairing status of the two bases r_1 and r_2 (if they exist) immediately to the right side of the base pair: stack means r_1 and r_2 form a base pair with each other, loop means that neither r_1 nor r_2 forms a base pair, end means that neither r_1 nor r_2 exists, stack+loop means that both r_1 and r_2 exist and one of the bases forms a base pair with another base while the other does not, stack+end means that only one of r_1 or r_2 exists and forms a base pair, loop+end means that only one of r_1 or r_2 exists and it does not form a base pair, and stack+stack means that both r_1 and r_2 exist and they both form base pairs with other bases. Stars indicate the possible continuation of the strands, which may be connected to other starred strands, provided the resulting complex is non-pseudoknotted.

3.2 The Arrhenius Kinetic Model

In our Arrhenius kinetic model, the activation energy of each transition depends on the immediate context of the closing or opening base pair. Our classification incorporates some, but not all, factors that may affect the activation energy of a transition. For example, the activation energy might depend on the strand sequence, but modeling this dependence would increase the number of free parameters, and we anticipate to have insufficient experimental evidence to accurately distinguish all relevant factors. However, we emphasize that transition rates in the model still depend on the nucleotide sequence via the nearest neighbor model of base pair stability that determines the free energy of a complex (see Eq. 2.19 and Eq. 2.21).

Consider a reaction where a base pair is formed or broken, and denote by $l, r \in \mathcal{C}$ one half of the local context on either side of the base pair. Our model differentiates between seven different half contexts

$$\mathcal{C} = \{\text{stack}, \text{loop}, \text{end}, \text{stack+loop}, \text{stack+end}, \text{loop+end}, \text{stack+stack}\} \qquad (3.1)$$

so that the set of local contexts is given by $\mathcal{C} \times \mathcal{C}$. The different half contexts are shown in Figure 3.1. In Appendix A.1, we show how to determine the local context of an elementary step transition, that is, the formation or breakage of a base pair.

The Arrhenius model is equal to the Metropolis model (Eq. 2.19, 2.20, 2.21), except that we now re-define $k_{\text{uni}} : \mathcal{C} \times \mathcal{C} \to \mathbb{R}_{>0}$ and $k_{\text{bi}} : \mathcal{C} \times \mathcal{C} \to \mathbb{R}_{>0}$ by setting

$$k_{\text{uni}}(l, r) = k_l k_r \qquad k_l = A_l \exp(-E_l/RT) \qquad k_r = A_r \exp(-E_r/RT) \tag{3.2}$$

$$k_{\text{bi}}(l, r) = \alpha k_{\text{uni}}(l, r) \tag{3.3}$$

where A_l, A_r are Arrhenius rate constants, E_l, E_r are activation energies, and α is a bimolecular scaling constant. We treat $\theta = \{\ln A_l, E_l \mid \forall l \in \mathcal{C}\} \cup \{\alpha\}$ as 15 free parameters that we fit to data.

3.3 Dataset

We compile a dataset of 376 experimentally determined reaction rate constants from the published literature for a wide range of DNA reactions, namely, hairpin closing, hairpin opening, helix association, helix dissociation, bubble closing, toehold-mediated three-way strand displacement, and toehold-mediated four-way strand exchange (Altan-Bonnet et al., 2003; Bonnet, 2000; Bonnet et al., 1998; Dabby, 2013; Kim et al., 2006; Machinek et al., 2014; Morrison and Stols, 1993; Reynaldo et al., 2000; Zhang and Winfree, 2009). Each data point in our dataset is annotated with a reaction temperature and the concentration of Na^+ and Mg^{2+} cations in the buffer. An overview of our dataset is given in Table 3.1.

As shown in Table 3.1, we partition the dataset into a training set of size 320, which we call $\mathcal{D}_{\text{train}}$, and a testing set with size 56, which we call $\mathcal{D}_{\text{test}}$. The training set covers a wide range of observations, in terms of both reaction types and half contexts. The testing set includes both unimolecular and bimolecular reactions.

Table 3.1: Dataset of experimentally measured reaction rate constants. The † sign indicates that the experiment was performed without Na^+ in the buffer, in which case our model computes the free energy as if 50 mM $[Na^+]$ is present (in addition to Mg^{2+}).

$\mathcal{D}_{\text{train}}$	$[Na^+]$ /M	$[Mg^{2+}]$ /mM	T /$^\circ C$	Source
Hairpin closing and opening	0.1		10–49	Figure 4 of Bonnet et al. (1998)
	0.1–0.5		10–49	Figure 6 of Bonnet et al. (1998)
	0.25		18–49	Figure 3.28 of Bonnet (2000)
	0.137		20	Figure 3 of Kim et al. (2006)
Bubble closing	0.1		25–45	Figure 4 of Altan-Bonnet et al. (2003)
Association and dissociation	1.0		4–68	Figure 6 of Morrison and Stols (1993)
	0.05†	4	30–55	Figure 6a of Reynaldo et al. (2000)
Toehold-mediated three-way strand displacement	0.05†	4	30–55	Figure 6b of Reynaldo et al. (2000)
	0.05†	12.5	25	Figure 3b of Zhang and Winfree (2009)
Toehold-mediated four-way strand exchange	0.05†	12.5	25	Table 5.2 of Dabby (2013)
$\mathcal{D}_{\text{test}}$				
Hairpin closing and opening	0.137		10–60	Figure 5a, b of Kim et al. (2006)
Toehold-mediated three-way strand displacement w/ mismatches	0.05†	10	23	Figure 2d of Machinek et al. (2014)

3.4 Modeling Framework

Given a parameterized kinetic model, a sufficient number of stochastic simulations could be run to estimate the model's prediction for an experimental reaction of interest. Unfortunately, obtaining small error bars on this estimate is prohibitively slow, and thus is not feasible within the inner loop of parameter inference procedures. To address this limitation, we developed a computational framework in which we obtain fast, exact predictions for a feasible approximation of the full Multistrand state space. Specifically, we use a simplified state space that is a strict subset of the full state space, enabling sparse matrix computations of MFPTs, from which reaction rate constants are predicted. With this computation in the inner loop, we used two methods for training the model. The first is a maximum a posteriori (MAP) approach that optimizes a single set of parameters, and the second is based on MCMC that produces an ensemble of parameter sets. In the latter case, a posterior parameter probability density is computed.

3.4.1 Simplified State Spaces

The number of states directly affects the computational cost of inference through the set of matrix equations (Eq. 2.6) that is solved for each reaction at each iteration of the parameter search. Therefore, in this chapter, for each reaction we use a subset of its full state space. For example, in this chapter, the largest simplified state space in the training data contains 14,438 states for a toehold-mediated four-way strand exchange reaction with more than 100 bases.

We generate a separate simplified state space $\mathcal{S}_r$ for each reaction r that we wish to model (Figure 2.1). To generate our simplified state spaces we define a set of rules as follows:

- We allow base pairs to form if and only if they occur in either the initial or target state of our simulation. For example, during the simulation of hairpin closing and hairpin opening, only base pairs that are consistent with the perfect alignment of the strand are permitted to form.

- We allow a maximum number of continuously hybridized domains for every type of reaction. For example, the states for hairpin closing and hairpin

opening contain at most one hybridized domain.

- We allow base pairs to form or break only at the edges of a continuous hybridized domain.

- We further prune the state space for reactions that still include a large number of states with our rules defined above, such as for toehold-mediated three-way strand displacement and toehold-mediated four-way strand exchange. Our additional heuristic rules for these reactions are explained in Appendix A.2.

With respect to our rules, we define the state space of a reaction r using a number of reaction-specific pointers $\langle p_0, p_1, ... \rangle \in \mathcal{S}_r$. The pointers indicate the begin and end points of the hybridized domains and can increase and decrease to change the size of the hybridized domain.

For states in hairpin closing and hairpin opening reactions in which we allow at most one hybridized domain, each state is represented by a tuple $\langle p_o, p_1 \rangle$, where $0 \leq p_0 \leq p_1 \leq m < l/2$. Here m is the length of the hybridized domain in the fully closed position and l is the length of the strand. The tuple indicates that the bases p_0 to $p_1 - 1$ are paired with bases $l - p_1$ to $l - p_0 - 1$, respectively, and no other base pairs are formed. The length of the hybridized domain is given by $p_1 - p_0$ and if $p_0 = p_1$ then the hybridized domain is absent in the given state. In each transition to a neighbor state, one of the pointers is incremented or decremented. For example, in Figure 3.2, the pointers for three states s, s', and s'' in hairpin closing and hairpin opening reactions are shown. Specifically, state s can transition to state s' by incrementing p_0 and it can transition to state s'' by decrementing p_1. Algorithm 1 shows the function `NeighborStates`(s) that generates the neighbor states of a hairpin state s in hairpin opening and hairpin closing reactions. In Appendix A.2, we describe `NeighborStates`(s) for helix association and dissociation, toehold-mediated three-way strand displacement, and toehold-mediated four-way strand exchange.

To enumerate the simplified state space for a reaction, we specify a reaction-specific `NeighborStates`(s), a reaction-specific set of initial states $\mathcal{S}_{\text{init}}$, and a reaction specific set of target states $\mathcal{S}_{\text{target}}$, and we use them in Algorithm 2. This

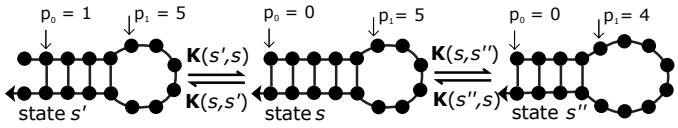


Figure 3.2: State s is defined by pointers $\langle p_0 = 0, p_1 = 5 \rangle$ and can transition to states s' and s'' defined by pointers $\langle p_1 = 0, p_1 = 5 \rangle$ and $\langle p_0 = 0, p_1 = 4 \rangle$, respectively.

algorithm uses a breadth-first search approach: initially, the queue Q and candidate state space $\mathcal{S}_{\text{init}}$ are composed of just the initial states. For every state in the queue, unexplored successor states are added to the candidate state space and then queued for exploration. In this algorithm, we use the same `NeighborStates`(s) function for reverse reactions, however we swap the initial and the target states. For example, in hairpin closing, the initial state ($\mathcal{S}_{\text{init}} = \{\langle 0,0 \rangle\}$) has no base pairs and the target state ($\mathcal{S}_{\text{target}} = \{\langle 0,m \rangle\}$) has m base pairs. For hairpin opening we swap these states. Note that our way of generating the simplified state spaces is semi-automatic, since we have to define `NeighborStates` for every type of reaction.

3.4.2 Estimating Mean First Passage Times

Given a parameterized kinetic model, for a reaction, we are interested in estimating its mean first passage time (MFPT). To do so, instead of running simulations that could take a long time to complete in the full state space of the reaction, we compute the MFPT of a truncated CTMC built from its simplified state space using Eq. 2.7. We derive the reaction rate constant of a unimolecular and a bimolecular reaction from its MFPT using Eq. 2.15 and 2.16, respectively.

3.4.3 Estimating the Unnormalized Posterior Distribution of the Parameters

Let θ be the set of parameters in a kinetic model. For a given experimentally observable reaction r, the predicted reaction rate constant $\hat{k}^r(\theta)$ will deviate from the experimental measurement k^r. We define the error of the prediction to be the $\log_{10}$ difference, $\varepsilon_r = \log_{10} k^r - \log_{10} \hat{k}^r(\theta)$. To produce a measure of likelihood

Algorithm 1: Generate the neighbor states of a hairpin state $s = \langle p_0, p_1 \rangle$ (see Figure 3.2)

Function `NeighborStates` ($s = \langle p_0, p_1 \rangle$)
- $\mathcal{N} \leftarrow \emptyset$
- **if** $\langle p_0, p_1 \rangle = \langle 0, 0 \rangle$ **then**
  - **for** $p \in [0, m-1]$ **do**
    - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1 \rangle$
- **else**
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1 \rangle \cup \langle p_0 + 1, p_1 \rangle \cup \langle p_0, p_1 - 1 \rangle \cup \langle p_0, p_1 + 1 \rangle$
- **foreach** $s' = \langle p'_0, p'_1 \rangle \in \mathcal{N}$ **do**
  - `// The state in which no base pair has formed is shown by` $\langle 0,0 \rangle$
  - **if** $p'_0 = p'_1$ and $p'_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0 \rangle$
- **foreach** $s' \in \mathcal{N}$ **do**
  - **if** !`AllowedState(`s'`)` **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$
- return $\mathcal{N}$

Function `AllowedState` ($s' = \langle p_0, p_1 \rangle$)
- **if** $!(0 \leq p_0 \leq p_1 \leq m)$ **then** return False
- return True

Algorithm 2: Generate state space

Function `GenerateStateSpace`
- $\mathcal{S} \leftarrow \mathcal{S}_{\text{init}}$, $Q \leftarrow \mathcal{S}_{\text{init}}$
- **while** $Q \neq \emptyset$ **do**
  - $\mathcal{N} \leftarrow \emptyset$
  - **foreach** $s \in Q$ **do**
    - **foreach** $s_p \in$ `NeighborStates`(s) **do**
      - **if** $s_p \notin \mathcal{S}$ **and** $s_p \notin \mathcal{S}_{\text{target}}$ **then**
        - $\mathcal{S} \leftarrow \mathcal{S} \cup s_p$
        - $\mathcal{N} \leftarrow \mathcal{N} \cup s_p$
  - $Q \leftarrow \mathcal{N}$
- $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}_{\text{target}}$
- return $\mathcal{S}$

for our parameter valuation, we assume ε_r is normally distributed with an unbiased mean and variance σ^2, so that $\varepsilon_r \sim N(0, \sigma^2)$. We treat σ as a nuisance parameter. For reaction r the likelihood function is given as

$$\mathrm{P}(r|\theta,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\left(\log_{10} k^r - \log_{10} \hat{k}^r(\theta)\right)^2 \Big/ 2\sigma^2 \right\}, \tag{3.4}$$

and the likelihood function over the set of training data is given as

$$\begin{aligned} \mathrm{P}(\mathcal{D}_{\text{train}}|\theta,\sigma) &= \prod_{r\in\mathcal{D}_{\text{train}}} \mathrm{P}(r|\theta,\sigma) \\ &= \exp\left\{ -\frac{\sum_{r\in\mathcal{D}_{\text{train}}} \left(\log_{10} k^r - \log_{10} \hat{k}^r(\theta)\right)^2}{2\sigma^2} - \frac{n}{2}\log 2\pi\sigma^2 \right\}, \end{aligned} \tag{3.5}$$

where n is the number of observations in $\mathcal{D}_{\text{train}}$. To define the probability of the parameters given the data we need to assume prior distributions for θ and σ. During preliminary fitting, a number of parameter values were found to be divergent, which we explain as follows. For a fixed temperature T and a fixed local context (l, r), there are many assignments of A_l, E_l and A_r, E_r that result in nearly equal transition rates $k_{\text{uni}}(l,r) = A_l A_r \exp\{-(E_l + E_r)/RT\}$ (we expand Eq. 3.2) that result in similar model predictions $\hat{k}^r(\theta)$. This allows dissimilar valuations for E and A to have nearly equal (log)likelihood scores (Eq. 3.5). The problem becomes even more apparent when we consider the intrinsic measurement error on k^r (for example, a standard deviation of 22% was reported by Machinek et al. (2014)), the limited range of temperatures (see Table 5.1) inherent to our observations, and the relative frequency of the different half contexts appearing in each simulation. In practice, $k_{\text{uni}}(l,r)$ is well constrained for many different $l, r \in \mathcal{C}$. As is common in data-fitting applications, we assume a regularization prior that improves the stability of the estimation. We assume that all parameters in θ are independent and identically Gaussian distributed with mean 0 and variance $\frac{1}{\lambda}$. In our inference, we use $\lambda = 0.02$, and the predictive quality of the model does not change for minor adjustments to λ. For the nuisance parameter σ, we use a non-informative Jeffreys prior (Jeffreys, 1946). Under these assumptions, the posterior distribution is

proportional to:

$$\begin{aligned}\mathrm{P}(\theta,\sigma|\mathcal{D}_{\text{train}}) &= \frac{\mathrm{P}(\mathcal{D}_{\text{train}}|\theta,\sigma)\mathrm{P}(\theta)\mathrm{P}(\sigma)}{\mathrm{P}(\mathcal{D}_{\text{train}})} \propto \mathrm{P}(\mathcal{D}_{\text{train}}|\theta,\sigma)\mathrm{P}(\theta)\mathrm{P}(\sigma) \\ &= \mathrm{P}(\mathcal{D}_{\text{train}}|\theta,\sigma)\left(\frac{2\pi}{\lambda}\right)^{-\frac{|\theta|}{2}} \exp\left\{-\frac{\lambda\|\theta\|_2^2}{2}\right\}\frac{1}{\sigma}. \end{aligned} \tag{3.6}$$

In conclusion, the log of the posterior distribution is equal to the following equation, up to an additive constant not depending on the parameters

$$\begin{aligned}&\log P(\theta,\sigma|\mathcal{D}_{\text{train}}) \approx \\ &\quad -(n+1)\log\sigma - \frac{1}{2\sigma^2}\sum_{r\in\mathcal{D}_{\text{train}}}\left(\log_{10}k_r - \log_{10}\hat{k}_r\right)^2 - \frac{\lambda}{2}\|\theta\|_2^2 \end{aligned} \tag{3.7}$$

where the squared $L2$ norm in Eq. 3.7 is computed as $\|\theta\|_2^2 = \alpha^2 + |\ln k_{\text{uni}}|^2 + |\ln k_{\text{bi}}|^2$ for the Metropolis model and as $\|\theta\|_2^2 = \alpha^2 + \sum_{l\in\mathcal{C}}|\ln A_l|^2 + \sum_{l\in\mathcal{C}}|E_l|^2$ for the Arrhenius model. Note that $|\theta| = 2$ for the Metropolis model and $|\theta| = 15$ for the Arrhenius model.

Our MAP approach seeks a unique parameter set that maximizes the normalized log posterior of the dataset (Eq. 3.7). We use the Nelder-Mead optimization method (Nelder and Mead, 1965), a gradient-free local optimizer. For MCMC, we use the *emcee* software package (Foreman-Mackey et al., 2013), that implements an affine invariant ensemble sampling algorithm.

3.5 Experiments

Here, we conduct computational experiments to evaluate the Arrhenius kinetic model. Our framework and the dataset, as well as an online appendix that has additional experimental plots and analysis, are available at https://github.com/DNA-and-Natural-Algorithms-Group/ArrheniusInference.

3.5.1 Experimental Setup

We fit the Metropolis and Arrhenius kinetic models using the MAP approach to a learn parameter set that maximizes Eq. 3.7. Using the MCMC approach, we

maximize the same equation, but instead obtain an ensemble of parameter sets.

The MAP method is sensitive to the initial parameters, and for the Metropolis model, we use $k_{\text{uni}} = 8.2 \times 10^6\ \text{s}^{-1}$ and $k_{\text{bi}} = 3.3 \times 10^5\ \text{M}^{-1}\text{s}^{-1}$, following known estimates for a one dimensional model of toehold-mediated strand displacement (Srinivas et al., 2013). For the Arrhenius model, we initialize $E_r = 3$ kcal mol^{-1} for all $r \in \mathcal{C}$ and we initialize α and A_r such that, at $T = 23^\circ C$, equally $k_{\text{uni}}(l,r) = 8.2 \times 10^6\ \text{s}^{-1}$ and $k_{\text{bi}}(l,r) = 3.3 \times 10^5\ \text{M}^{-1}\text{s}^{-1}$ for all local contexts $l, r \in \mathcal{C}$. For both models, we initialize $\sigma = 1$.

Results for the MCMC should generally depend less on the initial value of the sets in the ensemble. To initialize the parameter assignment for each parameter set in the MCMC ensemble, we realize random variables

$$\begin{aligned} E_r &\sim U(0,6) \times \text{ kcal mol}^{-1} & A_r &\sim U(0,10^4) \times s^{-1/2} & \forall r \in \mathcal{C} \\ k_{\text{uni}} &\sim U(0,10^8) \times \text{ s}^{-1} & k_{\text{bi}} &\sim U(0,10^8) \times \text{ M}^{-1}\text{s}^{-1} & \\ \alpha &\sim U(0,10) \times \text{M}^{-1} & \sigma &\sim U(0,1) & \end{aligned} \tag{3.8}$$

where $U(a,b)$ is the uniform distribution over (a,b). During the inference, the parameters are not restricted to initialization bounds, and instead we only require $k_{\text{uni}}, k_{\text{bi}}, A_l, \alpha$ and σ to be positive.

In the emcee software (Foreman-Mackey et al., 2013), an ensemble of walkers each represents a set of parameters, which are updated through *stretch moves*. Given two walkers θ_1 and θ_2, a new parameter assignment θ_1' for the first walker is generated as

$$\theta_1' = Z\theta_1 + (1-Z)\theta_2 \qquad g(Z=z) \propto \begin{cases} \frac{1}{\sqrt{z}} & \text{if } z \in \left[\frac{1}{a}, a\right] \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

where $g(z)$ is the probability density of Z. We use $a = 2$ (default value) and an ensemble of 100 walkers. We only use the last step of each walker to make predictions, which results in an ensemble of 100 parameter sets for each model.

For the MAP approach, we continue the inference until an absolute tolerance of 10^{-4} is reached. For the MCMC approach, we continue the inference until 750 iterations are performed per walker.

Table 3.2: Performance of the Metropolis and the Arrhenius models on the training and testing sets. The Mean Squared Error (MSE) is the mean of $|\log_{10} k^r - \log_{10} \hat{k}^r(\theta)|^2$ over $r \in \mathcal{D}$. The Within Factor of Three metric shows the percentage of reactions for which $|\log_{10} k^r - \log_{10} \hat{k}^r(\theta)| \leq \log_{10} 3$. Initial is the initial parameter set of the MAP approach (Section 3.5.1). MAP is the MAP inference method. Mode is the parameter set from the MCMC ensemble that has the highest posterior on $\mathcal{D}_{\text{train}}$. Ensemble is the MCMC ensemble method where the reaction rate constant $\hat{k}^r(\theta)$ is averaged over all parameter sets.

		Mean Squared Error		Within Factor of Three	
		$\mathcal{D}_{\text{train}}$	$\mathcal{D}_{\text{test}}$	$\mathcal{D}_{\text{train}}$	$\mathcal{D}_{\text{test}}$
Metropolis	Initial	.55	1.3	69.3%	33.9%
	MAP	.33	.94	79.0%	41.0%
	Mode	.33	.95	79.0%	41.0%
	Ensemble	.33	.99	79.6%	37.5%
Arrhenius	Initial	.59	1.3	71.2%	33.9%
	MAP	.14	.47	92.1%	73.2%
	Mode	.12	.40	92.8%	78.5%
	Ensemble	.12	.42	93.4%	78.5%

We implemented our framework in Python. All experiments were run on a system with 16 2.93GHz Intel Xeon processors and 64GB RAM, running openSUSE 42.1. On this system, each iteration takes less than 6 s.

3.5.2 Results

Table 3.2 shows the performance of the Metropolis and the Arrhenius models with the MAP and MCMC approaches. Appendix A.3 and Figure 3.3 plot the models' fitting and prediction for our training and testing sets. For details on computational settings for the approaches see Section 3.5.1. The Arrhenius model fits the training data better than the Metropolis model, which is unsurprising when considering the increase of adjustable parameters in the Arrhenius model (2 vs 15). However, the Arrhenius model also has better predictive qualities for the testing set, as evidenced by the MCMC ensemble mean standard deviation of $\sqrt{0.99} = 0.99$ for the Metropolis model and $\sqrt{0.42} = 0.64$ for the Arrhenius model. The improvement in the prediction of the testing set is apparent in Figure 3.3 and in Figures A.14 and A.15. In Figure 3.3, the models predict the Machinek et al. (2014) study of

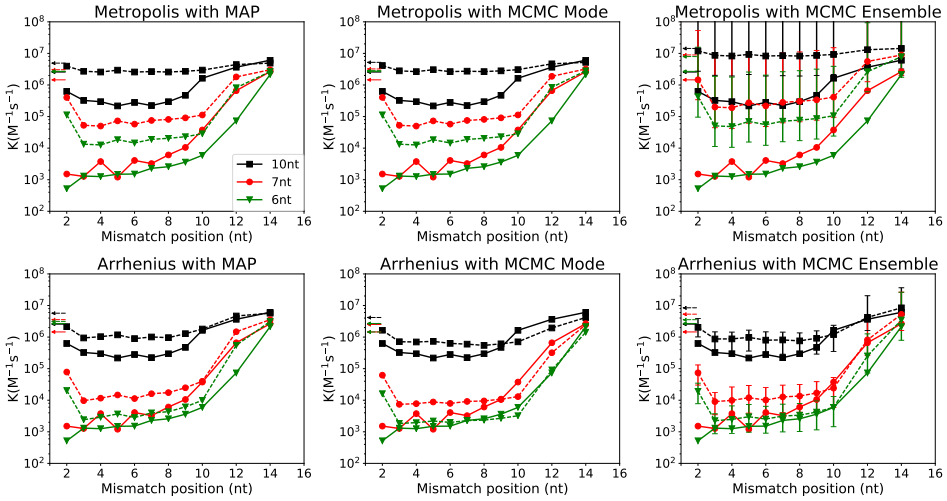


Figure 3.3: Model predictions (dashed lines) of reaction rate constants (y axis) for toehold-mediated three-way strand displacement with mismatches, experimental data (solid lines) from Figure 2d of Machinek et al. (2014). For the MCMC ensemble method, error bars indicate the range (minimum to maximum) of 100 predictions (see Section 3.5.1). Arrows indicate no mismatch. The mismatch in the invading strand affects the reaction rate. The length of the toehold domain is ten, seven, and six nucleotides long for ■, ●, and ▼, respectively.

toehold-mediated three-way strand displacement. It shows the effect of having a mismatch between the invader and the substrate in different positions and with different toehold domain lengths. Figures A.14 and A.15 correspond to the predictions of opening and closing rates for hairpins with short stems (1-2 nt). It is impressive that the models, when trained on a comprehensive training dataset, can predict the results of experiments not seen during training.

There are two reasons for the superior performance of the Arrhenius model. First, the presence of the activation energy allows the Arrhenius model to better calibrate to measurements at varying temperatures. On average, the reaction rate constants $k_{\text{uni}}(l,r)$ double in the Arrhenius model between $T = 25°C$ and $T = 60°C$ (this follows from the parameter values in which $\mathbb{E}[E_l + E_r] = 3.32$ kcal mol^{-1}). A second factor is the relation between the activation energy of a transition and the local context. In Figure 3.4, the inferred distribution of $k_{\text{uni}}(l,r)$ is given for all

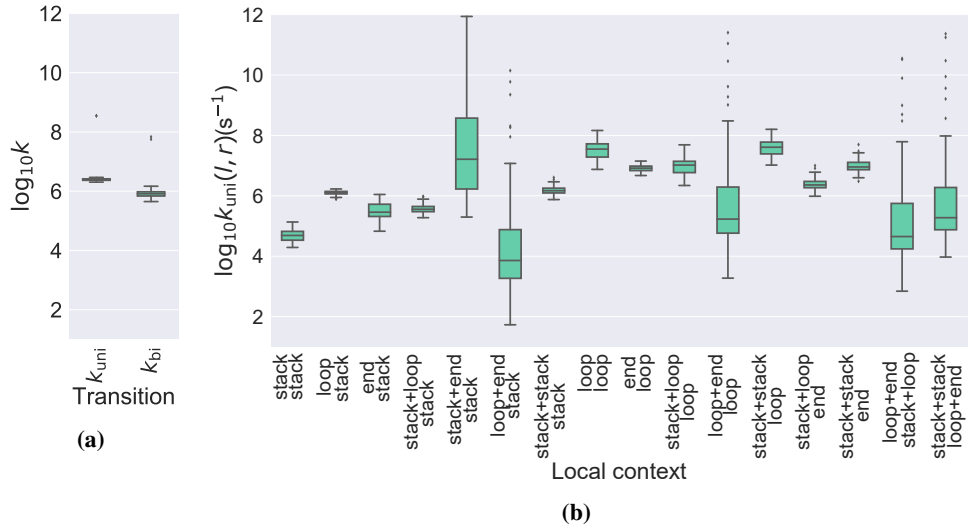


Figure 3.4: Box plots of model features inferred by the MCMC ensemble method, using a sample of 100 parameter sets. Edges of the box correspond to the first and third quartile of the distribution. The whisker length is set to cover all parameter values in the sample, or is limited to at most 1.5 times the box height with the outliers plotted separately. a) k_{uni} and k_{bi} for the Metropolis model. b) $k_{\text{uni}}(l,r)$ at $25°C$ for the Arrhenius model. Combinations that do not occur in the model are not shown.

local contexts that occur in the model. Strikingly, for many local contexts, the $k_{\text{uni}}(l,r)$ are narrowly distributed and often mutually exclusive, indicating that our model captures intrinsic qualitative differences in activation energy.

3.6 Summary and Directions for Future Improvements

In this chapter, we propose the Arrhenius kinetic model for interacting nucleic acid strands. The Arrhenius model is equal to the Metropolis model, except that transition rates depend on local contexts and activation energies. We show how to calibrate the model on various type of reactions using MFPT estimates. To facilitate MFPT estimation with matrix equations, we design simplified state spaces for various type of reactions with a semi-automatic approach. We conduct computational experiments on a dataset of a wide range of experimentally determined reaction

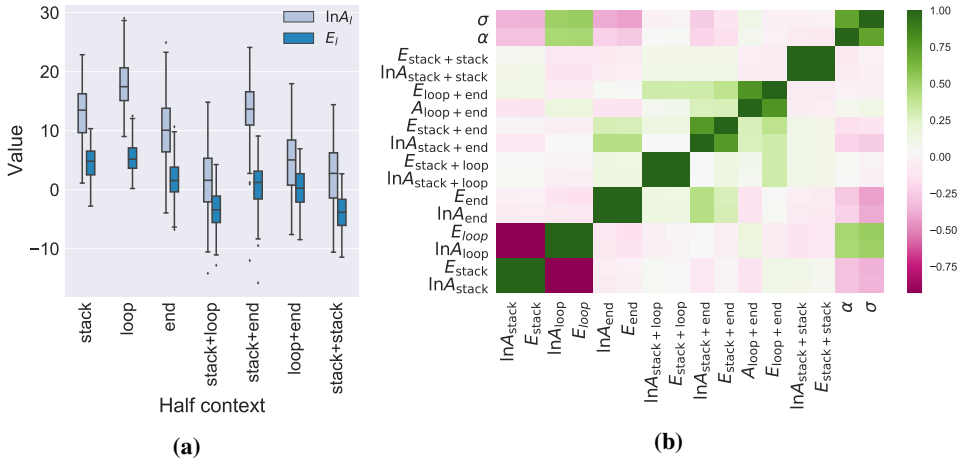


Figure 3.5: The Arrhenius model parameters inferred by the MCMC ensemble method. a) Box plots of the half context parameters. Edges of the box correspond to the first and third quartile of the distribution. The whisker length is set to cover all parameter values in the sample, or is limited to at most 1.5 times the box height with the outliers plotted separately. b) The Pearson correlation coefficients $R_{ij} = \frac{\text{cov}(\theta_i,\theta_j)}{\sigma_{\theta_i}\sigma_{\theta_j}}$, where $\text{cov}(X,Y) = \mathbb{E}[(X-\mathbb{E}[X])(Y-\mathbb{E}[Y])]$ and $\sigma_X = \sqrt{\mathbb{E}[(X-\mathbb{E}[X])^2]}$. (Color figure available online.)

rate constants to calibrate and evaluate the Arrhenius kinetic model. We show that the proposed Arrhenius kinetic model outperforms the Metropolis kinetic model.

A common problem for Arrhenius models in biophysics is that the limited range of temperatures in experimental data can result in ambiguous parameter inference, and this is indeed the case for our model with the current data set. Despite the generally narrow bands for the transition rates (Figure 3.4b), the inferred A and E parameters are poorly constrained, as is evident from the wide range in the parameter posterior probability distribution and correlation matrix (Figure 3.5). Mathematically, measurements at a single temperature only restrict $\ln A_l + \frac{-E_l}{RT}$ rather than A_l and E_l independently, and a significant fraction of the measurements were performed at constant temperature. If further mining of the existing experimental literature does not resolve the issue, one solution would be to develop customized experiments to calibrate the model further. Interestingly, the relative

lack of correlation between the parameters for different half contexts suggests that there could be benefit in subdividing the half context categories further.

We envision further improvements to the model by adjusting the state space and the thermodynamic energy model. For the state space, the requirement for hybridizing strands to only engage in perfectly aligned base pairing is not realistic, and methods such as pathway elaboration (introduced in Chapter 5) could alleviate these problems. Our simulation depends on the model of thermal stability implemented in the NUPACK software (Zadeh et al., 2011) and adjustments to the thermodynamic model also could improve the quality of our predictions. For example, hairpin closing rates are known to depend on the loop sequence, as open poly(A) loops are more rigid than poly(T) loops (Aalberts et al., 2003). The current thermodynamic model does not incorporate this effect, and we avoid comparing the model to measurements on poly(A) loop hairpins. Similarly, the initiation of branch migration is known to have a significant thermodynamic cost, with one study measuring a cost of 2.0 kcal mol^{-1} at room temperature (Srinivas et al., 2013). This initialization cost is not yet incorporated in NUPACK.

Finally, although our current analysis focuses on DNA, we believe our model and approach would also apply to RNA reaction kinetics.

Chapter 4

The Fixed Path Ensemble Inference Method

In the previous chapter, we compared estimated MFPTs with experimental MFPTs to calibrate kinetic models. To enable matrix computations for estimating MFPTs, we use a semi-automatic approach for obtaining simplified state spaces. However, this approach has two drawbacks: 1) The rules we defined, such as the requirement for hybridizing strands to only engage in perfectly aligned base pairing, may not be realistic; the MFPT of a reaction in its simplified state space may be noticeably different from the MFPT of the reaction in its full state space. 2) The simplified state spaces are customized for each type of reaction, making it difficult to generalize to other type of reactions. In this chapter, we seek a general approach that enables the rapid evaluation of perturbed parameters in parameter inference and also provides accurate MFPT estimates in the full state space of the reactions' CTMCs.

4.1 Introduction

In accurately predicting the MFPT of a reaction in the full state space of its CTMC, there are two challenging tasks. The first task is to estimate the MFPT of a reaction's CTMC, given a calibrated kinetic model. The second task is to calibrate parameters of kinetic models. These tasks are challenging because when nucleic acid strands interact, they are prone to the formation of many metastable secondary

structures due to stochastic formation and breakage of base pairs. The number of possible secondary structures nucleic acids can form may be exponentially large compared to the number of nucleotides the strands contain. Therefore, usually it is impossible to use exact matrix computations in the full state space. Moreover, to make accurate estimations with sampling methods, many sampled trajectories might be required, which might be time-consuming to obtain.

In this chapter, we address these tasks for reactions in their full state space of CTMCs. To estimate the MFPT of a reaction's CTMC, we show how to use a reduced variance stochastic simulation algorithm (RVSSA), a Rao-Blackwellised version (Lehmann and Casella, 2006) of SSA. In SSA, the variance of MFPT estimates arises for two reasons. First, the path to a target state affects the MFPT. Second, the holding time in each state affects the MFPT. RVSSA removes the stochasticity in the holding times by using expected holding times of states. We prove that RVSSA produces a lower variance estimator of the MFPT compared to SSA. Moreover, we show in our experiments that RVSSA has a lower variance than SSA in estimating the MFPT of a reaction's CTMC, when in the sampled paths there exists states that have large expected holding times. One interesting example that we identify is the association of poly(dA) and poly(dT) sequences in low concentrations (see Figure 4.1b).

To estimate parameters for nucleic acid kinetics modeled as CTMCs based on MFPTs, we show how to use a generalized method of moments (GMM) (Hansen, 1982) estimator. More importantly, we show how to use a fixed path ensemble inference (FPEI) approach that speeds up parameter estimation compared to a reference method that uses SSA directly during inference (SSAI). The GMM method is widely used in econometrics and has also been used in other fields including chemical reaction networks (Lück and Wolf, 2016). The GMM method can be used when a maximum likelihood estimate or a maximum a posteriori estimate is infeasible, as is the case with CTMCs that have very large state spaces. The GMM method minimizes a weighted norm of moment conditions obtained from samples. The moment conditions are functions of model parameters and the dataset such that the expectation of the moment conditions is zero at the true value of the parameters. To minimize the squared norm of the moment conditions, we use the Nelder-Mead direct-search optimization algorithm (Nelder and Mead, 1965), which has been fre-

quently used in optimization problems that have small stochastic perturbations in function values (Barton and Ivey Jr, 1996).

To speed up parameter estimation, we introduce and use FPEI. In this method, we first generate paths with SSA and then *condense* paths, where for every path, we compute the set of states and the number of times each state is visited. Rather than generating new trajectories with SSA for every parameter set variation (the SSAI method), in FPEI we use fixed condensed paths to speed up parameter estimation. For example, in this work, the length of the longest path is more than 1×10^8, whereas the number of unique states and transitions of the path is approximately 3.8×10^5 and 1.4×10^6, respectively. In FPEI, we use RVSSA to estimate the MFPT of the fixed paths given a new parameter set. Since the MFPT estimates obtained with fixed paths are biased, we alternate between minimizing the error of prediction on fixed paths, and resampling new paths and restarting the optimization method.

To implement RVSSA and FPEI, we augment the Multistrand kinetic simulator (Schaeffer, 2012; Schaeffer et al., 2015) (see Chapter 2). We conduct computational experiments on a dataset of 21 experimental DNA reactions that have moderate or large state spaces or are slow. The dataset consists of hairpin closing, hairpin opening, helix association, and helix dissociation with and without mismatches (Bonnet et al., 1998; Cisse et al., 2012; Hata et al., 2018; Wetmur, 1976). We compare the performance of RVSSA with SSA for MFPT estimation and FPEI with SSAI for parameter estimation. Results for our example data are encouraging, showing that FPEI speeds up parameter estimation compared to using SSAI, by more than a factor of three for slow reactions. Also, for reactions with large state spaces, it speeds up parameter estimation by more than a factor of two.

In Section 4.2, we describe the RVSSA method and introduce the FPEI method. In Section 4.3, we first describe the kinetic dataset that we use. Then we show how RVSSA performs for MFPT estimation and how FPEI works for parameter estimation.

4.2 Methodology

4.2.1 Mean First Passage Time Estimation

In SSA, the variance of MFPT estimates arises for two reasons. First, the path to a target state affects the MFPT. Second, the holding time in each state affects the MFPT. Hordijk et al. (1976) show how to obtain a reduced variance estimate of a steady-state measure of an irreducible and positive recurrent CTMC. Their constant holding-time method eliminates the variability in the random holding time of states and instead uses expected holding times. To estimate the MFPT of a reaction's CTMC, we formulate a Rao-Blackwellised version (Lehmann and Casella, 2006) of SSA, which similar to Hordijk et al. (1976) also eliminates the variability in the random holding times of states. However, the CTMC is not restricted to be irreducible or positive recurrent and the MFPT estimate is not necessarily a steady-state measure. We call this method the reduced variance stochastic simulation algorithm (RVSSA). Similar to SSA, RVSSA also produces a consistent and unbiased estimator of the MFPT when used within a Monte Carlo estimator, but has a smaller variance in predicting MFPTs compared to SSA[1].

In brief, in RVSSA, instead of sampling a random holding time for each state, we use an estimator based on the expected holding time. The algorithm proceeds as follows.

1. At a jump from the current state s_i, compute the expected holding time $\overline{T}_i$ before jumping to the next state, in other words, $\overline{T}_i = k_{s_i}^{-1} = (\sum_{s \in \mathcal{S}} k_{s_i s})^{-1}$. Note that $\mathbb{E}[\overline{T}_i \mid s_i] = k_{s_i}^{-1}$ and $\mathrm{Var}(\overline{T}_i \mid s_i) = 0$.

2. Step 2 of SSA (exactly as in Section 2.2): At a jump from the current state s_i, SSA samples the next state s_{i+1} from the outgoing transition probabilities of state s_i, in other words, $p(s_i, s) = \frac{k_{s_i s}}{k_{s_i}}, s_i \neq s$.

Let P be a path of length Z from state s to state t, with expected holding times $\overline{T}_1, ..., \overline{T}_{Z-1}$, obtained by using RVSSA with initial state s, and ending the first

[1] For our purpose here, we are only interested in the MFPT, so the smaller variance is good. In other contexts, the full distribution of FPTs will be of interest, and for that purpose only SSA, but not RVSSA, will be appropriate.

time that state t is sampled. In RVSSA, we compute the MFPT of the path as

$$Y^{\text{RVSSA}} = \sum_{i=1}^{Z-1} \overline{T}_i. \tag{4.1}$$

By using N independently sampled paths, we obtain a Monte Carlo estimator for the MFPT of the CTMC as $\hat{\tau}_N^{\text{RVSSA}} = \frac{1}{N}\sum_{n=1}^{N} Y_n^{\text{RVSSA}}$.

Theorem 1. *The estimator of the MFPT from state s to state t produced by RVSSA has a lower variance than the estimator produced by SSA.*

Proof. Let P denote a random path from state s to state t. We have $\mathbb{E}[F^{\text{SSA}} \mid P] = \mathbb{E}[Y^{\text{RVSSA}} \mid P]$, and consequently

$$\text{Var}(\mathbb{E}[F^{\text{SSA}} \mid P]) = \text{Var}(\mathbb{E}[Y^{\text{RVSSA}} \mid P]). \tag{4.2}$$

Also, $\mathbb{E}[\text{Var}(F^{\text{SSA}} \mid P)] > 0$, and $\mathbb{E}[\text{Var}(Y^{\text{RVSSA}} \mid P)] = \mathbb{E}[\text{Var}(\sum_{i=1}^{Z-1} \overline{T}_i \mid P)] = 0$ because $\overline{T}_i$ are constants and independent given P. Based on the law of total variance

$$\begin{aligned} &\text{Var}(Y^{\text{RVSSA}}) = \mathbb{E}[\text{Var}(Y^{\text{RVSSA}} \mid P)] + \text{Var}(\mathbb{E}[Y^{\text{RVSSA}} \mid P]) \overset{\text{by Eq. (4.2)}}{=} \\ &\mathbb{E}[\text{Var}(Y^{\text{RVSSA}} \mid P)] + \text{Var}(\mathbb{E}[F^{\text{SSA}} \mid P]) = \text{Var}(\mathbb{E}[F^{\text{SSA}} \mid P]) < \\ &\mathbb{E}[\text{Var}(F^{\text{SSA}} \mid P)] + \text{Var}(\mathbb{E}[F^{\text{SSA}} \mid P]) = \text{Var}(F^{\text{SSA}}). \end{aligned} \tag{4.3}$$

Therefore, it can be concluded that $\text{Var}(\hat{\tau}_N^{\text{RVSSA}}) = \text{Var}(\frac{1}{N}\sum_{n=1}^{N} Y_N^{\text{RVSSA}}) = \frac{1}{N}\text{Var}(Y^{\text{RVSSA}}) < \frac{1}{N}\text{Var}(F^{\text{SSA}}) = \text{Var}(\hat{\tau}_N^{\text{SSA}})$. □

For an unbiased estimator, the expected mean squared error (MSE) of the estimator is equal to the variance of the estimator (Wackerly et al., 2014). Consequently, RVSSA has a smaller MSE than SSA and requires fewer sampled paths to estimate the MFPT,

$$\mathbb{E}[(\hat{\tau}_N^{\text{RVSSA}} - \tau)^2] = \frac{1}{N}\text{Var}(Y^{\text{RVSSA}}) < \frac{1}{N}\text{Var}(F^{\text{SSA}}) = \mathbb{E}[(\hat{\tau}_N^{\text{SSA}} - \tau)^2]. \tag{4.4}$$

4.2.2 Parameter Estimation

In Section 4.2.1, we assume that the underlying parameters of the CTMCs are known. Here, we focus on estimating the underlying parameters of the transition rates when they are not known a priori.

To estimate model parameters, we formulate a generalized method of moments (GMM) (Hansen, 1982) objective function based on experimental and predicted MFPTs. The GMM estimators have desirable statistical properties under suitable conditions, such as consistency and asymptotic normality. The GMM method minimizes a weighted norm of moment conditions. The moment conditions are functions of model parameters and observed values such that the expectation of the moment conditions is zero at the true value of the parameters. Given a column vector $\mathbf{g}$ of moment conditions and its transpose $\mathbf{g}^{\mathrm{T}}$, the GMM method seeks the true parameter set θ^* as

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathbf{g}(\theta)^{\mathrm{T}} \mathbf{W} \mathbf{g}(\theta), \tag{4.5}$$

where $\mathbf{W}$ is a positive-definite matrix that controls the variance of the estimator. For optimally chosen weights, which depend on the covariance of the moment conditions at the true parameter set θ^*, the estimator has the smallest possible variance for the parameters. Since the true parameter set is unknown, there exist several approaches to deal with this issue. For example, the two-step GMM estimator (Hansen et al., 1996) uses the identity matrix in the first step to estimate a parameter set. In the second step, it uses the estimated parameters to produce the weighting matrix and reestimates the parameters. In our experiments, we only use the identity weighting matrix, which produces a consistent and asymptotic normal GMM estimator, and leave other options to future work.

Let θ be a parameter set for a kinetic model that parameterizes the CTMC of reactions, and let θ^* be the true parameter set. For reaction r, based on the experimental MFPT τ^r and an unbiased estimator of the MFPT $\hat{\tau}^r$, we can define a moment condition as $g^r(\theta) = \hat{\tau}^r(\theta) - \tau^r$. However, since reactions occur at timescales that cover many orders of magnitude, from slow reactions, such as helix dissociation, to faster reactions, such as hairpin closing, and since we are using an

identity matrix, we use $\log_{10}$ differences instead; we define a moment condition as

$$g^r(\theta) = \log_{10} \hat{\tau}^r(\theta) - \log_{10} \tau^r, \tag{4.6}$$

where we approximate $\mathbb{E}[g^r(\theta^*)] = \mathbb{E}[\log_{10} \hat{\tau}^r(\theta^*)] - \log_{10} \tau^r \approx 0$ for the true parameter set θ^* (if one exists). This approximation is reasonable for an unbiased and low variance estimator of the experimental MFPT τ^r. The Taylor expansion of $\mathbb{E}[\log_{10} \hat{\tau}^r(\theta^*)]$ around $\log_{10} \mathbb{E}[\hat{\tau}^r(\theta^*)] = \log_{10} \tau^r$ is $\mathbb{E}[\log_{10} \hat{\tau}^r(\theta^*)] \approx \mathbb{E}\left[\log_{10} \tau^r + \frac{1}{\tau^r}(\hat{\tau}^r - \tau^r) - \frac{1}{2(\tau^r)^2}(\hat{\tau}^r - \tau^r)^2\right] = \log_{10} \tau^r - \frac{\mathrm{Var}(\hat{\tau}^r(\theta^*))}{2(\tau^r)^2}$, where the second term disappears. Also, note that based on Eqs. 2.15 and 2.16, instead of Eq. 4.6 we equivalently use $g^r(\theta) = \log_{10} \hat{\tau}^r(\theta) - \log_{10} \tau^r = \log_{10} k^r - \log_{10} \hat{k}^r(\theta)$, which is commonly used in related work (Zhang et al., 2018; Zolaktaf et al., 2017). Based on the entire reactions of the dataset $\mathcal{D}$, we define the GMM estimator as

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{r \in \mathcal{D}} \left(\log_{10} k^r - \log_{10} \hat{k}^r(\theta)\right)^2. \tag{4.7}$$

This can be recognized as the least mean squared error (MSE) parameter set.

In our experiments (described in Section 4.3.3), we seek a parameter set that minimizes the GMM estimator. However, we also considered using the negative of Eq. (14) from our previous work (Zolaktaf et al., 2017), where $g^r(\theta)$ is defined to be normally distributed with an unbiased mean and variance σ^2, and a small *L2* regularization term is also defined. With this objective function, the predictive quality of the fixed path ensemble inference (FPEI) approach, which we describe later on, only slightly changes for our dataset.

To minimize the objective function, we use the Nelder-Mead direct-search optimization algorithm (Nelder and Mead, 1965). To approximate a local optimum parameter set θ with size $|\theta|$, the algorithm maintains a simplex of $|\theta| + 1$ parameter sets. The algorithm evaluates the function value at every parameter set of the simplex. It proceeds by attempting to replace a parameter set that has the worst function value with a parameter set reflected through the centroid of the remaining $|\theta|$ parameter sets in the simplex with expansion and contraction as needed. The algorithm uses only the ranks of the function values to determine the next parameter set, and therefore has been frequently used in optimization problems that have

small stochastic perturbations in function values (Barton and Ivey Jr, 1996). This robustness is essential for its use in SSAI.

In SSAI, during the optimization, to obtain an unbiased estimate of τ^r for every parameter set variation, we use SSA. However, obtaining new trajectories for every parameter set is computationally expensive. One reason is that transitions might be repeatedly sampled. Therefore the length of a trajectory could be much larger than the number of unique states and transitions of the trajectory (see Section 4.3.3). We propose to use FPEI which uses an ensemble of fixed paths, with an efficient data structure, to speed up parameter estimation. In FPEI, for every reaction, we build a fixed set of paths with an initial parameter set θ_0. For a new parameter set θ_m, we use the fixed paths to estimate the MFPT. To speed up computations, we condense paths; for every path, we compute the set of states and the number of times each state is visited. We compute the holding time of a state in a path as if the path is regenerated in the full state space. To compute the holding time of a state under a new parameter set, we need to compute the total outgoing transition rate from the state under the new parameter set. Therefore, we also store information about the outgoing neighbors of the states that affect the outgoing transition rate. Alternatively, depending on memory and storage limitations, similar to SSA and RVSSA, we could repeatedly compute the outgoing neighbors of the states on the fly. Given this data, as the parameter set is updated to θ_m, we compute the MFPT of path P according to RVSSA as

$$Y^{\text{FPEI}}(\theta_m) = \sum_{i=1}^{Z-1} \overline{T}_i(\theta_m), \text{ where } \overline{T}_i(\theta_m) = \frac{1}{k_{s_i}(\theta_m)}, \tag{4.8}$$

where the transition rates of the CTMC depend on the parameter set θ_m and the path is obtained with θ_0. Because of the condensed representation, this formula is not literally computed, but rather a mathematically equivalent one with fewer terms is computed. Given N fixed paths obtained with θ_0, we estimate the MFPT of the CTMC that is parameterized with θ_m as $\hat{\tau}_N^{\text{FPEI}}(\theta_m) = \frac{1}{N}\sum_{n=1}^{N} Y_n^{\text{FPEI}}(\theta_m)$.

With fixed paths, the MFPT estimates are biased and the learned parameter set might not perform well in the full state space where other paths are possible. Therefore, to reduce the bias and to ensure that the ensemble of paths is a fair

Algorithm 3: SSAI

$\theta \leftarrow \theta_0$ `// Choose initial parameter set` θ_0
Initialize the simplex in the Nelder-Mead algorithm using θ and its perturbations
while stopping criteria not met **do**
  `// See Section 4.3.3 for our stopping criteria`
  $\theta \leftarrow$ Retrieve a parameter set from the Nelder-Mead algorithm
  Update the free parameters of the kinetic model with θ
  foreach reaction $r \in$ dataset $\mathcal{D}$ **do**
    foreach n=1,2,...,N **do**
      Sample a trajectory P_n using SSA and calculate its FPT using Eq. 2.12
    Calculate the MFPT of the reaction using the FPTs of the trajectories
  Calculate the GMM function in Eq. 5.11 using the MFPT of the reactions
  Update the simplex in the Nelder-Mead algorithm based on the GMM function

sample with respect to the optimized parameters, we alternate between minimizing the error of prediction on fixed paths, and resampling new paths and restarting the optimization method. An overview of our parameter estimation framework using SSAI and FPEI, with a GMM estimator and the Nelder-Mead algorithm, is given in Algorithm 3 and Algorithm 4, respectively.

We also considered a normalized importance sampling approach (Doucet and Johansen, 2009), to obtain consistent estimators of the MFPTs using fixed paths. In this approach, we also compute the set of traversed transitions and how often each of those transitions occur in the path. We weigh the estimated MFPT of each path P by the relative likelihood of the path given the new and the initial parameter sets $\tilde{L}(\theta_m) = \frac{L(\theta_m)}{L(\theta_0)}$, where $L(\theta_m)$ is the likelihood of P under parameter assignment θ_m. For RVSSA, $L(\theta_m) = \prod_{i=1}^{Z-1} \frac{k_{s_i s_{i+1}}(\theta_m)}{\sum_{s\in\mathcal{S}} k_{s_i s}(\theta_m)} e^{-\sum_{s\in\mathcal{S}} k_{s_i s}(\theta_m)\overline{T}_i(\theta_m)}$, and we estimate the MFPT as $\hat{\tau}_N^{\text{FPEI}}(\theta_m) = \frac{1}{\sum_{n=1}^{N}\tilde{L}_n(\theta_m)} \sum_{n=1}^{N} \tilde{L}_n(\theta_m) Y_n^{\text{FPEI}}(\theta_m)$. In our experiments, this alternate normalized importance sampling approach performed poorly, since the effective sample size of the relative likelihoods was small.

Algorithm 4: FPEI

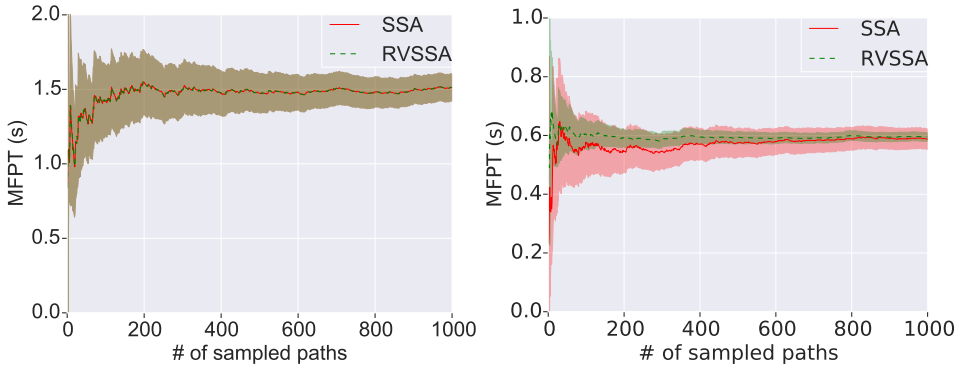
```
θ ← θ_0 // Choose initial parameter set θ_0
while stopping criteria not met do
    // See Section 4.3.3 for our stopping criteria
    Update the free parameters of the kinetic model with θ
    foreach reaction r ∈ dataset 𝒟 do
        foreach n=1,2,...,N do
            Sample a path P_n using RVSSA
            Condense path P_n for the reaction
    Initialize the simplex in the Nelder-Mead algorithm using θ and its
     perturbations
    while stopping criteria not met do
        θ ← Retrieve a parameter set from the Nelder-Mead algorithm
        Update the free parameters of the kinetic model with θ
        foreach reaction r ∈ dataset 𝒟 do
            foreach n=1,2,...,N do
                Calculate the MFPT of path P_n using Eq. 4.8
            Calculate the MFPT of the reaction using the MFPTs of the paths
        Calculate the GMM function in Eq. 5.11 using the MFPT of the
         reactions
        Update the simplex in the Nelder-Mead algorithm based on the GMM
         function
```

4.3 Experiments

Here, we conduct computational experiments to evaluate the RVSSA and FPEI methods. We implement FPEI on top of the Multistrand kinetic simulator. Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/FPEI.

4.3.1 Dataset

We use 21 experimentally determined reaction rate constants published in the literature for DNA reactions of hairpin closing, hairpin opening, helix association, and helix dissociation with and without mismatches (Bonnet et al., 1998; Cisse et al., 2012; Hata et al., 2018; Wetmur, 1976). Each reaction of the dataset is annotated with a temperature and the concentration of Na^+. The dataset is summarized in



(a) Reaction no. 16 **(b)** Reaction no. 20

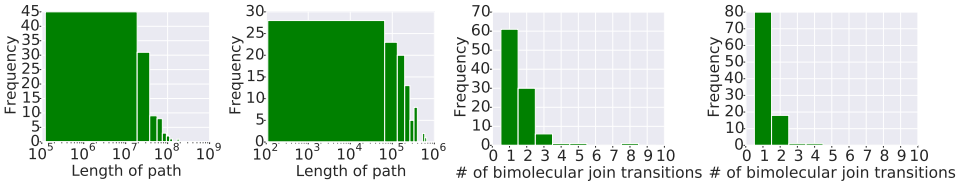
Figure 4.1: The MFPT and 95% confidence interval of SSA and RVSSA, where the kinetic model is parameterized with θ_0. In both (**a**) and (**b**), RVSSA and SSA are using the same sampled paths. In (**a**), RVSSA and SSA have similar variance. The average computation time per sampled path, defined as the total computation time divided by the total number of sampled paths, is 3×10^2 s. In (**b**), RVSSA has a lower variance than SSA. The average computation time per sampled path is 0.5 s.

Table 4.1.

For a hairpin opening reaction, we define the initial state to be the system microstate in which a strand has fully formed a duplex and a loop. We define the target state to be the system microstate in which the strand has no base pairs. Hairpin closing is the reverse reaction, where a strand with no base pair forms a fully formed duplex and a loop. For a helix dissociation reaction, we specify the initial state to be the system microstate in which two strands have fully formed a helix. We define the set of target states to be the set of system microstates in which the strands have detached and there are no base pairs within one of the strands. Helix association is the reverse bimolecular reaction. We define the target state to be the microstate in which the duplex has fully formed.

4.3.2 Mean First Passage Time Estimation

Figure 4.1a and Figure 4.1b show the performance of RVSSA compared with SSA for helix association reactions no. 16 and 20, respectively. To sample paths and



(a) Reaction no. 16 **(b)** Reaction no. 20 **(c)** Reaction no. 16 **(d)** Reaction no. 20

```
1:        ..................... ..................... t=0.000000000 s
2:        ..................... ..(....)............. t=0.000000045 s
.
19166996: ...........(......).... ..................... t=0.751824960 s
19166997: ..................... ..................... t=0.751825005 s
19166998: ..................(...+.....)................. t=0.751825051 s
19166999: ..................(...+.....).......(.....)... t=0.751825095 s
.
19167442: ((((((((((((((((((((((+))))))))))))))))))))))) t=0.751841627 s
```

(e) Reaction no. 16

```
1:      ..................... ..................... t=0.000000000 s
2:      ...........(...........+....)................. t=0.219236868 s
3:      ...........(.(.........+..).)................. t=0.219236888 s
4:      ...........(...........+....)................. t=0.219236898 s
.
173450: ((((((((((((((((((((((+))))))))))))))))))))) t=0.222540278 s
```

(f) Reaction no. 20

Figure 4.2: Histogram of the length of 100 random paths obtained with RVSSA for (**a**) reaction no. 16 and (**b**) reaction no. 20. Histogram of the number of bimolecular join transitions of the random paths for (**c**) reaction no. 16 and (**d**) reaction no. 20. Snapshot of the i-th state visited, dot-parentheses notation and jump times for a random path obtained with RVSSA for (**e**) reaction no. 16 and (**f**) reaction no. 20. The jump time at the i-th state is equal to the jump time at the $(i-1)$-th state plus the holding time of the $(i-1)$-th state. The green highlighting indicates where a bimolecular step occurs.

Table 4.1: Dataset of experimentally determined reaction rate constants. The concentration of the strands is set to 1×10^{-8} M, 5×10^{-8} M, and 1×10^{-8} M, for reactions no. 1-15, 16-19, and 20-21, respectively.

Reaction Type	**No.**	**Sequences**	**T** /$^{\circ}C$	**[Na]$^{+}$** /**M**	$\log_{10} k^r$	**Source**
Hairpin closing	1-5	CCCAA-$(T)_{30}$-TTGGG	14.4-29.8	0.1	3.53-3.69	Figure 4 of Bonnet et al. (1998)
Hairpin opening	6-10	CCCAA-$(T)_{30}$-TTGGG	14.4-29.8	0.1	2.14-3.30	Figure 4 of Bonnet et al. (1998)
Helix dissociation (with a mismatch)	11-15	AGGACTTGT + ACAAGACCT AGGACTTGT + ACAAGTGCT AGGACTTGT + ACAAGTCGT AGGACTTGT + ACAAGTCCA AGGACTTGT†	37	0.2	0.19-0.92	Figure S4 of Cisse et al. (2012)
Helix association	16-19	GCCCACACTCTTACTTATCGACT† GCACCTCCAAATAAAAACTCCGC† CGTCTATTGCTTGTCACTTCCCC† ACCCTTTATCCTGTAACTTCCGC†	25	0.195	5.71-6.68	Table 1 of Hata et al. (2018)
Helix association	20-21	25-mer Poly(dA)† 25-mer Poly(dG)†	48-78	0.4	-	Table 1 of Wetmur (1976)

† The complement of the demonstrated sequence is also a reactant.

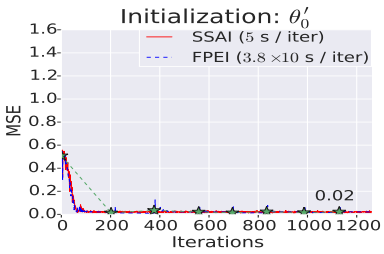
trajectories, we parameterize the kinetic model with the Metropolis initial parameter set (Srinivas et al., 2013; Zolaktaf et al., 2017), in other words, $\theta_0 = \{k_{\text{uni}} = 8.2 \times 10^6\ \text{s}^{-1},\ k_{\text{bi}} = 3.3 \times 10^5\ \text{M}^{-1}\text{s}^{-1}\}$. In both Figure 4.1a and Figure 4.1b, RVSSA and SSA have the same paths, but the algorithms generate different holding times for the states of the paths. In Multistrand's implementation of SSA, the effort needed to sample the holding time in the current state is small when compared to the task of computing outgoing transition rates. In Figure 4.1a, RVSSA and SSA perform the same, whereas in Figure 4.1b, RVSSA has a lower variance than SSA, consistently. To understand the discrepancy between the two figures, we analyze the experiments, described below.

In Figure 4.2a and Figure 4.2b, the average length of the paths for both reaction no. 16 and reaction no. 20 is large. Also, in Figure 4.2c and Figure 4.2d, both reactions have a small number of bimolecular transitions on average. In Figure 4.2e, for reaction no. 16, the state where two strands are disconnected has a small holding time, because the state has many fast unimolecular transitions between complementary bases within a strand in addition to the slow bimolecular transitions. However, in Figure 4.2f, for reaction no. 20, the state where the two strands are disconnected has a large holding time, since there are no complementary bases within a poly(dA) or poly(dT) strand and the only transitions are slow bimolecular transitions. RVSSA has a significantly lower variance for reaction no. 20 compared to SSA, because in the sampled paths, there exists states that have large expected holding times. SSA has a large variance in generating holding times for these states. Overall, in our experiments with parameter set θ_0, RVSSA has a lower variance than SSA for reactions no. 20 and 21, but performs similar to SSA for other reactions in Table 4.1.

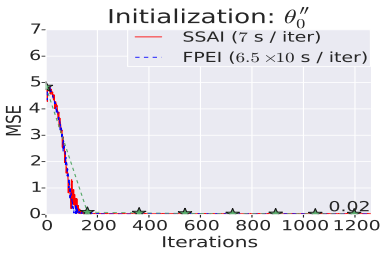
4.3.3 Parameter Estimation

Figure 4.3 shows the MSE, defined as the mean of $|\log_{10} k^r - \log_{10} \hat{k}^r(\theta)|^2$ on different reactions, of FPEI and SSAI over various iterations, where the methods are learning parameters for the Arrhenius kinetic model (Zolaktaf et al., 2017). Also, it shows the average computation time per iteration, defined as the total computation time divided by the total number of iterations. Figure 4.4 shows the MSE and average computation time per iteration when the entire dataset is used. Reactions no. 20-21 are excluded in parameter estimation because of our uncertainty in our interpretation of the reported measurements. For reactions no. 1-15, FPEI and SSAI use 200 paths and 200 trajectories, respectively. For reactions no. 16-19, where simulations are more time-consuming, FPEI and SSAI use 20 paths and 20 trajectories, respectively.

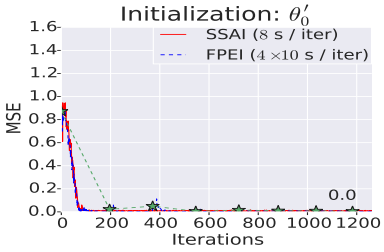
We conduct distinct experiments by starting with two sets of initial parameters, where paths and trajectories are generated in a reasonable time. In one group of experiments (Figs. 4.3a, 4.3c, 4.3e, 4.3g, and Figure 4.4a), we initialize the simplex in the Nelder-Mead algorithm with the Arrhenius initial parameter set (Srinivas et al.,



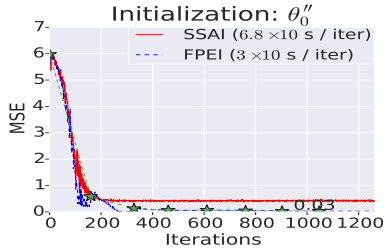
(a) Reactions no. 1-5 (hairpin closing)



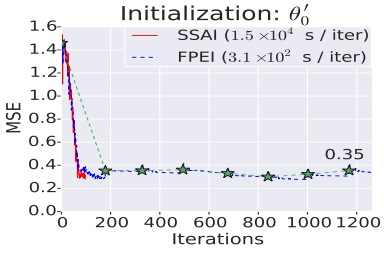
(b) Reactions no. 1-5 (hairpin closing)



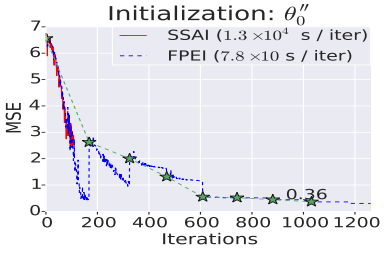
(c) Reactions no. 6-10 (hairpin opening)



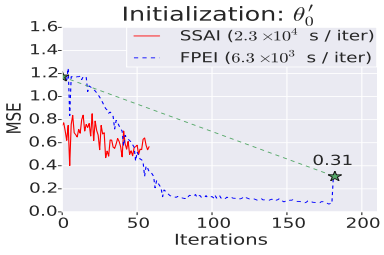
(d) Reactions no. 6-10 (hairpin opening)



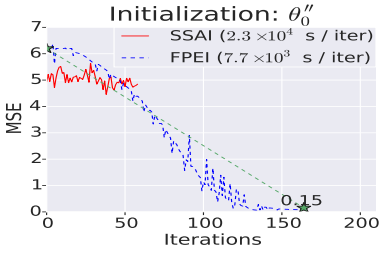
(e) Reactions no. 11-15 (helix dissociation)



(f) Reactions no. 11-15 (helix dissociation)

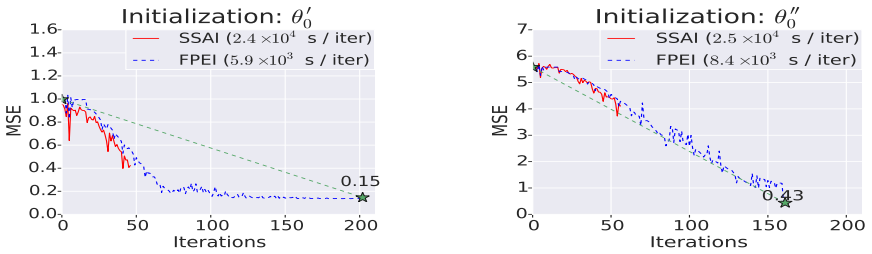


(g) Reactions no. 16-19 (helix association)



(h) Reactions no. 16-19 (helix association)

Figure 4.3: The MSE of SSAI and FPEI on different types of reactions from Table 4.1. The average computation time per iteration is shown in the label of each method. The ★ markers show the MSE when trajectories are rebuilt from scratch using the learned parameter set from FPEI. In Figs. 4.3e- 4.3h, the SSAI traces stop at earlier iterations than the FPEI traces, even though SSAI was allocated more time than FPEI.



(a) Reactions no. 1-19 **(b)** Reactions no. 1-19

Figure 4.4: As in Figure 4.3, but reactions no. 1-19 are all used as the dataset.

2013; Zolaktaf et al., 2017), in other words, $\theta_0' = \{A_l = 468832.1058\ \text{s}^{-1/2}, E_l = 3\ \text{kcal mol}^{-1} \mid \forall l \in \mathcal{C}\} \cup \{\alpha = 0.0402\ \text{M}^{-1}\}$ and its perturbations (in each perturbation, a parameter is multiplied by 1.05). In FPEI, we also use θ_0' to generate fixed paths. In another set of experiments (Figs. 4.3b, 4.3d, 4.3f, 4.3h, and Figure 4.4b), we adapt parameter set $\theta_0'' = \{A_l = 468832.1058\ \text{s}^{-1/2}, E_l = 2\ \text{kcal mol}^{-1} \mid \forall l \in \mathcal{C}\} \cup \{\alpha = 0.0402\ \text{M}^{-1}\}$ from θ_0' to increase the initial MSE in all experiments. We initialize the simplex in the Nelder-Mead algorithm with θ_0'' and its perturbations (in each perturbation, a parameter is multiplied by 1.05). In FPEI, we also generate fixed paths with θ_0''. In SSAI, we run the optimization until a limit on the number of iterations is reached or until a time limit is reached, which ever comes first. We also use this as the first stopping criteria in FPEI. In FPEI, to reduce the bias and to ensure that the ensemble of paths is a fair sample with respect to the optimized parameters, occasionally, the fixed paths are rebuilt from scratch and the optimization restarts. To this end, we set the second stopping criteria in FPEI to 200 iterations or 200 function evaluations of the Nelder-Mead algorithm, whichever comes first. Note that this empirical value is subject to change for different experiments. We could improve the method, by investigating a more robust way of when to update the paths. For example, we could compare the performance of SSA with the fixed paths in shorter intervals and update the fixed paths when their predictive quality diverges from SSA. During the optimization, we use an infinite value for parameter sets that have rates that are too slow or too fast; we bound downhill unimolecular and bimolecular rates (Eq. (7) and Eq. (8) of (Zolaktaf et al., 2017)) in $[1 \times 10^4, 1 \times 10^9]\ \text{s}^{-1}$ and in $[1 \times 10^4, 1 \times 10^9]\ \text{M}^{-1}\text{s}^{-1}$, respectively.

In Figs. 4.3d-4.3h, FPEI reaches a minimal MSE more quickly than SSAI; consider the average computation time per iteration multiplied by the number of iterations to reach a minimal MSE. However, in Figs. 4.3a-4.3c, SSAI reaches a minimal MSE more quickly than FPEI. This is because in Figs. 4.3d-4.3h, the number of unique states is significantly smaller than the length of the paths. For example, in Figure 4.3h, in the first set of fixed paths, the average length of a path is more than 2.3×10^7, whereas the average number of unique states and transitions is less than 1.5×10^5 and 5.6×10^5, respectively. In Figure 4.3a, the average length of a path is 4.6×10^2, whereas the average number of unique states and transitions is 1.3×10^2 and 2.4×10^2, respectively. In Figs. 4.3e-4.3f, which are slow dissociation reactions, compared to SSAI, FPEI speeds up parameter estimation by more than a factor of three. In Figs. 4.3g-4.3h, compared to SSAI, FPEI speeds up parameter estimation by more than a factor of two. Also, the speed of FPEI in all the figures could be improved with a better implementation of the method; in our implementation, in the first iteration, computing neighbor states of all states in a fixed condensed path is slow, whereas the later iterations which reuse the fixed condensed paths are much faster than SSAI.

In Figure 4.4a and Figure 4.4b, where reactions no. 1-19 are all used in the optimization, FPEI speeds up parameter estimation, by more than a factor of two compared to SSAI. In Figure 4.4a, FPEI reaches an MSE of 0.15 in 1.2×10^6 s, whereas SSAI reaches an MSE of 0.39 in the same time. In Fig 4.4b, FPEI reaches an MSE of 0.43 in 1.3×10^6 s, whereas SSAI reaches an MSE of 3.72 in the same time.

4.4 Summary and Directions for Future Improvements

In this chapter, we show how to use RVSSA to estimate the MFPT of a reaction's CTMC. In our experiments, RVSSA has a lower variance than SSA in estimating the MFPT of a reaction's CTMC, when in the sampled paths there exists states that have large expected holding times. Furthermore, we show how to use FPEI along with a GMM estimator and the Nelder-Mead algorithm to estimate parameters for nucleic acid kinetics modeled as CTMCs. In FPEI, we use RVSSA instead of SSA, since the MFPT estimator produced by RVSSA has a lower variance. In FPEI, we

use fixed condensed paths because sampling new paths for every parameter set is computationally expensive. Since using fixed paths leads to biased estimates, we alternate between minimizing the error of prediction on fixed paths, and resampling new paths and restarting the optimization method. FPEI speeds up computations when the number of unique states is significantly smaller than the length of sampled paths. In our experiments on a dataset of DNA reactions, FPEI speeds up parameter estimation compared to using SSAI, by more than a factor of three for slow reactions. Also, in our experiments, for reactions with large state spaces, it speeds up parameter estimation by more than a factor of two.

We used MFPT estimates obtained from FPEI to find a local optimimum parameter set. Alternatively, to approximate the posterior distribution of parameters, we could use MFPT estimates obtained from FPEI along with an Approximate Bayesian Computation (ABC) (Jennings and Madigan, 2017; Marin et al., 2012) approach to likelihood-free inference.

FPEI can be applied to reactions modeled as CTMCs, when the fixed paths can be produced in a timely manner. Generating paths for FPEI could be computationally expensive for rare reactions, such as helix dissociation from Morrison and Stols (Morrison and Stols, 1993). The runtime also depends on the kinetic model and its parameterization. It would be helpful to make FPEI applicable for such reactions, by speeding up the generation of the fixed paths.

Finally, we evaluated FPEI in the context of DNA reactions. It would be useful to adopt and evaluate FPEI in other CTMC kinetic models, and other domains that require estimating MFPTs in CTMCs, such as protein folding.

Chapter 5

The Pathway Elaboration Method

In the previous chapter, we addressed MFPT estimation and the rapid evaluation of perturbed parameters in the full state space of reactions' CMTCs. However, the methods investigated are not suitable for reactions that happen on a long time scale, that is rare events, in large state spaces. The reason is that the paths are generated according to SSA which could be time-consuming. Therefore, in this chapter, we propose a time-efficient probabilistic approach which can be used to estimate the MFPT of reactions in large state spaces, including rare events, and also enables the rapid evaluation of perturbed parameters.

5.1 Introduction

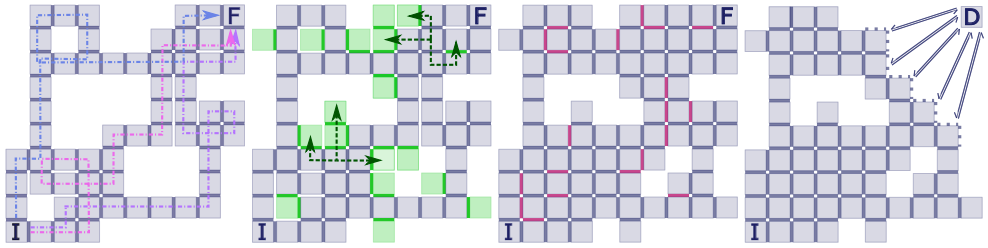
The speed at which nucleic acids interact is difficult to predict in their full state space of CTMCs. The number of secondary structures interacting nucleic acid strands may form is exponentially large in the length of the strands. For example, in the dissociation of a duplex consisting of a 25-mer poly(C) strand and a 25-mer poly(G) strand there are at least $\Omega(2^{25})$ secondary structures that can occur. Typical to interacting nucleic acid strand reactions are high energy barriers that prevent the reaction from completing, meaning that long periods of simulation time are required before successful reactions occur. Consider reactions that occur with

rates lower than $10\ \mathrm{s}^{-1}$ or $10000\ \mathrm{M}^{-1}\ \mathrm{s}^{-1}$ such as dissociation of long duplexes and three-way strand displacement at room temperature (see Table 5.1). These types of reactions are slow to simulate not because the simulator takes longer to generate trajectories for larger molecules, but the slowness is instead a result of the energy landscape: at low temperatures, duplexes simply are more stable, and require longer simulated time until their dissociation is observed.

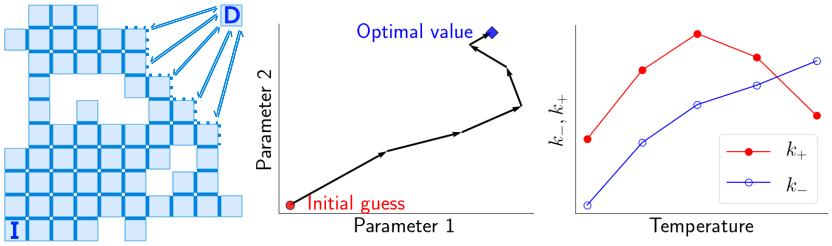
Here we are interested in a method that successfully addresses all three challenges for CTMCs which we described in the Chapter 1: large state spaces, rare events, and efficient recomputation for perturbed model parameters. We explore this possibility by developing a method which uses biased and local stochastic simulations to build truncated state spaces relevant to a (possibly rare) event of interest. We demonstrate that our method is suitable for predicting nucleic acid kinetics.

We propose the *pathway elaboration* method for estimating MFPTs of detailed-balance CTMCs. Pathway elaboration is a time-efficient probabilistic truncation-based approach which can be used for MFPT estimation of rare events and also enables the rapid evaluation of perturbed parameters. In pathway elaboration, we first construct a pathway by biasing SSA simulations from the initial states to the targets states. The biased simulations are guaranteed to reach the target states in expected time that is linear in the distance from initial to target states. Then, we expand the pathway by running SSA simulations for a limited time from every state of the pathway, with the intention of increasing accuracy by increasing representation throughout the pathway. Finally, we compute all possible transitions between the sampled states that were not encountered in the previous two steps. For the resulting truncated CTMC, we solve a matrix equation to compute the MFPT to the target state (or states). Since solving matrix equations could be slow for large CTMCs, pathway elaboration includes a δ-pruning step to efficiently prune CTMCs while keeping MFPT estimates within predetermined upper bounds. In this way, solving the system for other parameter settings becomes faster. Figure 5.1 illustrates a conceptual figure of the pathway elaboration method and its applications.

To evaluate pathway elaboration, we focus on prediction of nucleic acid kinetics. We implement the method using the Multistrand kinetic simulator (Schaeffer,



(a) Pathway construction **(b)** State elaboration **(c)** Transition construction **(d)** δ-pruning



(e) Updating perturbed parameters **(f)** Parameter estimation **(g)** Perturbed temperature

Figure 5.1: The pathway elaboration method and its applications. Pathway elaboration makes possible MFPT estimation of rare events and the rapid evaluation of perturbed parameters. Here, in the underling detailed-balance CTMC, boxes in a square grid represent states of the CTMC, with transitions between adjacent boxes, initial state I at bottom left and target state F at top right. (**a**) From state I, sample paths that are biased towards the target state F. Three sampled paths are shown with blue, pink and purple dotted lines. (**b**) From each sampled state found in the previous step, run short unbiased simulations to fill in the neighborhood. Simulations from two states are shown with green dashed lines. The green states and transitions are sampled. (**c**) Include all missing transitions between the states that were sampled in steps a and b. The red transitions are included. (**d**) Prune states that are expected to reach the target state quickly by redirecting their transitions into a new state. (**e**) For perturbed model parameters, keep the topology of the truncated CTMC, but update the rates of the transitions and the probability distribution of the states. (**f**) We can use truncated CTMCs for perturbed parameters. For example, we can use them for estimating model parameters. (**g**) As another example, we can use them to predict forward (k_+) and reverse (k_-) reaction rate constants as the temperature changes.

2013; Schaeffer et al., 2015) (see Chapter 2). The challenges of large state spaces, rare events, and handling perturbed parameters all arise for nucleic acid kinetics. Since the number of secondary structures may be exponentially large in the length of the strands, applying matrix equations is infeasible. Also, SSA often takes a long time to complete for rare nucleic acid reactions. Moreover, to calibrate the underlying kinetic model or to obtain a desired functionality the rapid evaluation of perturbed parameter is required (see Figures 5.1f and 5.1g). We conduct computational experiments on a dataset containing 267 experimentally determined kinetics of interacting nucleic acid strands (Bonnet et al., 1998; Cisse et al., 2012; Hata et al., 2018; Machinek et al., 2014; Zhang et al., 2018). The dataset consists of various types of reactions – namely, hairpin opening, hairpin closing, helix dissociation with and without mismatches, helix association, and toehold-mediated three-way strand displacement with and without mismatches – for which experimentally measured rate constants vary over 8.6 orders of magnitude. We partition the 267 reactions into two sets, 237 where SSA is feasible for MPFT estimation, i.e., completes within two weeks, and the remaining 30 for which SSA is not feasible.

In our experiments, first we use pathway elaboration to gain insight on the kinetics of two contrasting strand displacement reactions from Machinek et al. (Machinek et al., 2014), one being a rare event. Then, to evaluate the estimations of pathway elaboration, we compare them with estimations obtained from SSA for the 237 feasible reactions that were feasible with SSA. We use SSA since obtaining MFPTs with matrix equations is not possible for many of these reactions and SSA provides statistically correct trajectories. We find that for the settings we use, the mean absolute error (MAE) of the $\log_{10}$ reaction rate constant (or equivalently the MAE of the $\log_{10}$ MFPT) is 0.13. This is a reasonable accuracy since the $\log_{10}$ reaction constant predictions of SSA vary over 7.7 orders of magnitude (see Figure 5.6). In our experiments, pathway elaboration is on average 5 times faster than SSA on these reactions. We further use pathway elaboration to rapidly evaluate perturbed model parameters during optimization of Multistrand kinetic parameters. We use the same 237 reactions for training the optimizer and the remaining 30 as our testing set. Using the optimized parameters, pathway elaboration estimates of rate constants on our dataset are greatly improved over the estimates using

non-optimized parameters. For the training set, the MAE of the $\log_{10}$ reaction rate constants of pathway elaboration with experimental measurements reduces from 1.43 to 0.46, that is, a 26.9-fold error in the reaction rate constant reduces to a 2.8-fold error on average. The MAE over the 30 remaining reactions – which involve rare events and require large state spaces – reduces from 1.13 to 0.64, that is, a 13.4-fold error in the reaction rate constant reduces to a 4.3-fold error on average. On average for these 30 reactions, pathway elaboration takes less than two days, whereas SSA is not feasible within two weeks. The entire optimization and evaluation takes less than five days.

In Section 5.2 we present the pathway elaboration method. Afterwards in Section 5.3, we evaluate pathway elaboration for nucleic acid kinetics.We start by describing our kinetic dataset and our experimental setup that is common in all of our experiments. Then, we analyze and compare a truncated CTMC for a toehold-mediated three-way strand displacement reaction with a truncated CTMC for a toehold-mediated three-way strand displacement reaction with a mismatch, where we build the truncated CTMCs with pathway elaboration. After that, we compare the performance of pathway elaboration with SSA on a wide range of reactions. Finally, we use pathway elaboration for the rapid evaluation of perturbed parameters in parameter estimation.

5.2 Methodology

We are interested in the efficient simulation of rare events in detailed-balance CTMCs and also the rapid evaluation of mildly perturbed parameters. Our approach is to create a reusable in-memory representation of CTMCs, which we call a truncated CTMC, and to compute the MFPTs through matrix equations (see Eqs. 2.7 and 2.8).

We propose the pathway elaboration method for building a truncated detailed-balance CTMC $\hat{\mathcal{C}}^R$ for a detailed-balance CTMC $\mathcal{C}^R$. We call this approach the pathway elaboration method as we build a truncated CTMC by elaborating an ensemble of prominent paths in the system. The method has three main steps to build a truncated CTMC, and an additional step for the rapid evaluation of perturbed parameters.

Algorithm 5: The pathway elaboration method.

Function `PathwayElaboration(`$\mathcal{C}^R$`,`N`,`β`,`K`,`κ,π'`)`

$(\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}}, \pi) = \mathcal{C}^R$

$\mathcal{S}_0 \leftarrow$ `ConstructPathway(`$\mathcal{C}^R$`,`N`,`β`,`π'`)`

$\hat{\mathcal{S}} \leftarrow \emptyset$

for $s \in \mathcal{S}_0$ **do**

$\mathcal{S}' \leftarrow$ `ElaborateState(`$s,\mathcal{C}^R$`,K,`κ`)` `// Run SSA` K `times from` s `with a time limit of` κ `and return the visited states.`

$\hat{\mathcal{S}} \leftarrow \hat{\mathcal{S}} \cup \mathcal{S}'$

$\hat{\mathbf{K}} \leftarrow$ Construct rate matrix from $\hat{\mathcal{S}}$ and $\mathbf{K}$ `// Eq. 2.9.`

return $\hat{\mathcal{C}}^R = (\hat{\mathcal{S}}, \hat{\mathbf{K}}, \hat{\pi}_0, \hat{\mathcal{S}}_{\text{target}}, \hat{\pi})$ `// For` $\hat{\pi}_0$ `and` $\hat{\pi}$`, see Eq. 2.10 and Eq. 2.11, respectively.`

Function `ConstructPathway(`$\mathcal{C}$`,`N`,`β`,`π'`)`

$(\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}}) = \mathcal{C}$

$\mathcal{S}_0 \leftarrow \emptyset$

for n = 1 to N **do**

Sample $s \sim \pi_0$

$\mathcal{S}_0 \leftarrow \mathcal{S}_0 \cup \{s\}$

Sample $s_b \sim \pi'$

for t =1,2, ... **do**

if $s = s_b$ **then** break

Sample $z \sim \text{Uniform}(0,1)$

if $z < \beta$ **then** `// Bias simulations towards` s_b `using Eq. 5.1.`

Sample $s'|s \sim \mathbf{P}(\cdot|X_{t-1} = s)$

else

Sample $s'|s \sim \breve{\mathbf{P}}_{s_b}(\cdot|X_{t-1} = s)$

$\mathcal{S}_0 \leftarrow \mathcal{S}_0 \cup s'$

$s \leftarrow s'$

return $\mathcal{S}_0$

1. The first step, "pathway construction", uses biased simulations to find an ensemble of short paths from the initial states to the target states. This step is inspired by importance sampling (Andrieu et al., 2003; Doucet and Johansen, 2009; Hajiaghayi et al., 2014; Madras, 2002; Rubino and Tuffin, 2009) and exploration-exploitation trade-offs (Sutton and Barto, 2018).

2. The second step, "state elaboration", uses SSA from every state in the pathway to add additional states to the pathway, with the intention of increasing accuracy. This step is inspired by the string method (Weinan et al., 2005).

3. The third step, "transition construction", creates a matrix of transitions between every pair of states obtained from the first and second steps.

4. The fourth step, "δ-pruning", prunes the CTMC obtained from the previous steps to enable the rapid evaluation of perturbed parameters.

These steps results in a t truncated detailed-balance CTMC $\hat{\mathcal{C}}^R = (\hat{\mathcal{S}}, \hat{\mathbf{K}}, \hat{\pi}_0, \hat{\mathcal{S}}_{\text{target}}, \hat{\pi})$, where $\hat{\pi}_0$ and $\hat{\pi}$ are obtained via renormalization from π_0 and π. Figure 5.1, parts (**a**) through (**d**), illustrates the key steps of the pathway elaboration method, and Algorithm 5 provides high-level pseudocode. We next describe these steps in detail.

Pathway construction. We construct a pathway by biasing N SSA simulations towards the target states. We bias a simulation by using the shortest-path distance function $d : \mathcal{S} \times \mathcal{S}_{\text{target}} \to \mathbb{R}_{\geq 0}$ from every state $s \in \mathcal{S}$ to a fixed target state $s_b \in \mathcal{S}_{\text{target}}$ (Hajiaghayi et al., 2014; Kuehlmann et al., 1999). For every biased path, we can use a different s_b. Therefore, in general, we can sample s_b from a probability distribution π' over the target states. Given s_b, we use an exploitation-exploration trade-off approach. At each transition, the process randomly chooses to either decrease the distance to s_b or to explore the region based on the actual probability matrix of the transitions.

Let $\mathcal{D}_{s_b}(s)$ be the set of all neighbors of s whose distance with s_b is one less than the distance of s with s_b, and let $\mathbf{P}(s, s')$ be as in Eq. 2.1. Instead of sampling states according to $\mathbf{P}$, we use $\tilde{\mathbf{P}} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$ where

$$\tilde{\mathbf{P}}(s,s') =$$
$$\begin{cases} \mathbf{P}(s,s') = \frac{\mathbf{K}(s,s')}{\sum_{s''\in\mathcal{S}}\mathbf{K}(s,s'')} & 0 \le z \le \beta, \\ \breve{\mathbf{P}}_{s_b}(s,s') = \frac{\mathbf{K}(s,s')\mathbf{1}\{s'\in\mathcal{D}_{s_b}(s)\}}{\sum_{s''\in\mathcal{S}}\mathbf{K}(s,s'')\mathbf{1}\{s''\in\mathcal{D}_{s_b}(s)\}} & \beta < z \le 1. \end{cases} \tag{5.1}$$

Here z is chosen uniformly at random from $[0,1]$, β is a threshold, and $\mathbf{1}\{.\}$ is an indicator function that is equal to 1 if the condition is met and 0 otherwise. When $\beta = 1$, then $\tilde{\mathbf{P}}(s,s') = \mathbf{P}(s,s')$.

Proposition 1. *Let d_{max} be the maximum distance from a state in a CTMC to target state s_b. Then when $0 \le \beta < 1/2$, the expected length of a pathway that is sampled according to Eq. 5.1 is at most $\frac{d_{max}}{1-2\beta}$.*

Proof. Based on the distance of states with s_b, we can project a biased path that is generated with Eq. 5.1 to a 1-dimensional random walk R, where coordinate $x = 0$ corresponds to s_b and coordinate $x > 0$ corresponds to all states $s \neq s_b$ with $d(s,s_b) = x$. From the definition of $\tilde{\mathbf{P}}$ and since all states have a path to s_b by a transition to a neighbor state that decreases the distance by one, at each step, the random walk either takes one step closer to $x = 0$ with probability at least $1-\beta$ or one step further from $x = 0$ with probability at most β. If we let $E(R,k)$ denote the expected time for random walk R to reach 0 from k, then we have that when $0 \le \beta < 1/2$,

$$E(R,k) \le \frac{k}{1-2\beta}, \tag{5.2}$$

which follows from classical results on biased random walks—see Feller XIV.2 (Feller, 1968). Therefore, if $0 \le \beta < 1/2$, the proposition holds, and the state space built with N biased paths from the initial state s_0 to a target state s_b has expected size

$$\mathbb{E}[|\hat{\mathcal{S}}|] \le \frac{N \cdot d(s_0,s_b)}{1-2\beta} \le \frac{N \cdot d_{\max}}{1-2\beta}. \tag{5.3}$$

If for each biased path, the initial state is sampled from π_0 and the target state is sampled from π', then we sum over the N sampled (initial state, target state) pairs, and the total expected state space size is still bounded by $\frac{N \cdot d_{\max}}{1-2\beta}$. □

For efficient computations, we should be able to compute the shortest-path distance efficiently. For elementary step models of interacting nucleic acid strands, we can compute $d(s, s_b)$ by computing the minimum number of base pairs that need to be deleted or formed to convert s to s_b. Multistrand provides a list of base pairings for every complex microstate in a system microstate (state) and we can calculate the distance between two states in a running time of O(b), where b is the total number of bases in the strands.

State elaboration. By using Eq. 5.1, a biased path could have a low probability of reaching a state that has a high probability of being visited with SSA. For example, in some helix association reactions from Zhang et al. (Zhang et al., 2018), intra-strand base pairs are likely to form before completing hybridization. However, the corresponding states do not lie on the shortest paths from the initial states to the target states. Let c be the minimum number of transitions from s_0 that are required to reach s but which increase the distance to s_b. Now, let the random walk R be defined as the previous section. Let P_1 denote the probability of reaching s_b before reaching s for this random walk. Following classical results on biased random walks (Feller, 1968), for $\beta \neq 1/2$,

$$P_1 \geq \frac{(\frac{\beta}{1-\beta})^c - 1}{(\frac{\beta}{1-\beta})^{d_{s_b}(s_0)+c} - 1}. \tag{5.4}$$

In the extreme case if $\beta = 0$, then $P_1 = 1$ and the probability of reaching s will be 0.

Not including states that are likely to be visited with SSA could lead to inaccurate MFPT estimates. For example, assume for three states s, I and F, there exists reversible transitions between I and F and between I and s but there are no transitions between s and F. Let π be a probability distribution over the states that satisfies the detailed balance conditions with the rate matrix $\mathbf{K} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$. Assume $\pi(I)$ is low, whereas $\pi(F) = \pi(s)$ are high. Also, assume $\mathbf{K}(I,s) = \mathbf{K}(I,F)$ is large, and that $\mathbf{K}(s,I) = \mathbf{K}(F,I)$ is small. Therefore, in the actual CTMC, starting from I the process has an equal chance of transitioning to either s or F. Upon transition to s it is expected to have a large holding time, which increases the MFPT to

F. However, if we use $\beta = 0$ for generating a biased path from I to F, it will never reach s and the MFPT to F will be an underestimate of the actual MFPT.

Therefore, for detailed-balance CTMCs, we elaborate the pathway to possibly include states that have a high probability of being visited with SSA but were not included with our biased sampling. Here, we use SSA to elaborate the pathway; we run K simulations from each state of the pathway for a maximum simulation time of κ, meaning that a simulation stops as soon as the simulation time becomes greater than κ. By simulation time we mean the expected time of a SSA trajectory, not the wall-clock time. K and κ are tuning parameters that affect the quality of predictions. The running time of elaborating the states in the pathway with this approach is $O(|\hat{\mathcal{S}}|K\kappa k_{\max})$, where $\hat{\mathcal{S}}$ is the state space of the pathway and $k_{\max}$ is the fastest rate in the pathway. An alternate approach is to run each simulation for a number of transitions instead of a simulation time. Another approach is to add all states that are within distance r of every state of the pathway. However, with this approach, the size of the state space could explode, whereas by using SSA the most probable states will be chosen.

Note that any elaboration which stops before hitting the target state might be problematic for non-detailed-balance CTMCs. Trajectories that stop while visiting a state for the first time might effectively be introducing a spurious sink into the enumerated state space. Without reversibility that last transition of the elaboration might be irreversible. Sink states that are not a target state make the MFPT to the target states infinite. For example in Figure 5.2, assume in the elaboration step, the simulation finds s and s' but not s'' (or any other neighbor of s'). Then without the reversible transition, s' will be a sink state and the MFPT to the target state F will be infinite. Moreover, having reversible transitions that do not obey the detailed balance condition may make MFPT estimations large. For example, in Figure 5.2 assume that the reversible transitions between s and s' do not obey detailed balance. Also, assume $\pi(s)$ and $\pi(s')$ are both high, and that $\mathbf{K}(s,s')$ is large whereas $\mathbf{K}(s',s)$ is small. Therefore, if the elaboration stops at s' it will make the MFPT large. However, in the full state space, s' might quickly reach F through a fast transition to s''. Thus, the state elaboration step may not be suitable for non-detailed-balance CTMCs.

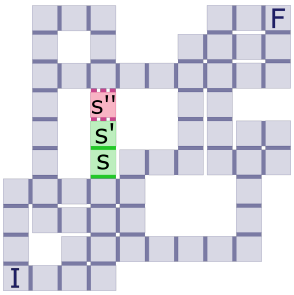


Figure 5.2: In the elaboration step, the simulation finds s and s' but not s''. Without detailed balance, a slow transition from s' to s could make the MFPT to F large. However, in the full state space, s' might quickly reach F through a fast transition to s''.

Transition construction. After the states of the pathway are elaborated, fast transitions between the states of the pathway could still be missing. To make computations more accurate, we further compute all possible transitions in $\hat{\mathcal{S}}$ that were not identified in the previous two steps. In related roadmap planning work (Kavraki et al., 1996; Tang et al., 2005; Thomas et al., 2013), states are connected to their nearest neighbors as identified by a distance metric. We can include all missing transitions by checking whether every two states in $\hat{\mathcal{S}}$ are neighbors in $O(|\hat{\mathcal{S}}|^2)$ or by checking for every state in $\hat{\mathcal{S}}$ whether its neighbors are also in $\hat{\mathcal{S}}$ in $O(|\hat{\mathcal{S}}|m\})$, where m is the maximum number of neighbors of the states in the original CTMC.

δ-pruning. Given a (truncated) CTMC in which we can compute the MFPT from every state to the target state, one question is: which states and transitions can be removed from the Markov chain without changing the MFPT from the initial states significantly? This question is especially relevant for the rapid evaluation of perturbed parameters, where MFPTs need to be recomputed often.

Given a CTMC $\mathcal{C} = (\mathcal{S}, \mathbf{K}, \pi_0, \mathcal{S}_{\text{target}})$ and a pruning bound δ, let τ_s denote the MFPT from state s to $\mathcal{S}_{\text{target}}$ and let τ_{π_0} denote the MFPT from the initial states to $\mathcal{S}_{\text{target}}$. Let $\mathcal{S}_{\delta p} = \{s \in \mathcal{S} \mid \tau_s < \delta\tau_{\pi_0} \text{ and } \pi_0(s) = 0\}$ be the set of states that are δ-close to $\mathcal{S}_{\text{target}}$ and that are not an initial state. We construct the δ-pruned CTMC $\mathcal{C}_\delta = (\mathcal{S}_\delta, \pi_0, \mathbf{K}_\delta, \{s_d\})$ over the pruned set of states $\mathcal{S}_\delta = \mathcal{S} \setminus \mathcal{S}_{\delta p} \cup \{s_d\}$, where s_d is the new target state. For $s, s' \in \mathcal{S}_\delta \setminus \{s_d\}$, we update the rate matrix

$\mathbf{K}_\delta : \mathcal{S}_\delta \to \mathbb{R}_{\geq 0}$ by $\mathbf{K}_\delta(s, s_d) = \sum_{s' \in \mathcal{S}_{\delta_p}} \mathbf{K}(s, s')$ and $\mathbf{K}_\delta(s, s') = \mathbf{K}(s, s')$. Note that $\mathbf{K}_\delta(s_d, s)$ is not used in the computation of the MFPT (see Eq. 2.7), so we can simply assume $\mathbf{K}_\delta(s_d, s) = 0$. Alternatively, to retain detailed-balance conditions, we can define the energy of s_d as $E(s_d) = -RT \log \sum_{s'' \in \mathcal{S}_{\delta_p}} e^{-\frac{E(s'')}{RT}}$ (see Eqs.7.1 and 7.2 from Schaeffer (2013)) and define $\mathbf{K}_\delta(s_d, s) = e^{-\frac{E(s)-E(s_d)}{RT}} \mathbf{K}_\delta(s, s_d)$.

For the pruned CTMC $\mathcal{C}_\delta = (\mathcal{S}_\delta, \pi_0, \mathbf{K}_\delta, \{s_d\})$, let the MFPT $\tau^\delta_{\pi_0}$ be given as usual (Eq. 2.8). Then by construction

$$\tau^\delta_{\pi_0} \leq \frac{\tau_{\pi_0}}{1+\delta}. \tag{5.5}$$

We can calculate the MFPT from every state to the target states by solving Eq. 2.7 once for CTMC $\mathcal{C}$. Therefore, the running time of δ-pruning depends on the running time of the matrix equation solver that is used for Eq. 2.7. For a CTMC with state space $\mathcal{S}$, the running time of a direct solver is at most $O(|\mathcal{S}|^3)$. Recently, nearly-linear time algorithms have been developed for directed Laplacian systems, which are applicable to the generative matrices of CTMCs (Cohen et al., 2018). For iterative solvers the running time is generally less than $O(|\mathcal{S}|^3)$. After the equation is solved, the CTMC can be pruned in $O(|\mathcal{S}|)$ for any δ. Note that for a given bound δ, the runtime for solving Eq. 2.7 for the pruned CTMC $\mathcal{C}_\delta$ might still be high. In that case, a larger value of δ is required. To set δ in practice, it could be useful to consider the number of states that will be pruned for a given δ, that is $|\mathcal{S}_{\delta_p}|$.

Updating perturbed parameters. We are interested in rapidly estimating the MFPT to target states given mildly perturbed parameters. Our approach is to reuse a truncated CTMC for mild perturbations. The MFPT estimates will be biased in this way. However, we could have significant savings in running time by avoiding the cost of sampling from scratch. We would still have to solve Eq. 2.7, but with efficient solvers (Virtanen et al., 2020) the cost could be less than the cost of sampling from scratch. For example in Table 5.2, on average, solving the matrix equation is faster than SSA by a factor of 47 and is faster than building the truncated CTMC by a factor of 10. Moreover, it might be possible to reduce the cost of solving matrix equations by reusing calculations from previous equations (Brand, 2006; Bunch and Nielsen, 1978; Parks et al., 2006). We do not take advantage of these methods

for solving matrix equations in this work, but we still obtain significant speed-ups by reusing truncated CTMCs.

A perturbed thermodynamic model parameter affects the energy of the states. Therefore, to update the transition rates, we would also have to recompute the energy of the states. A perturbed kinetic model only affects the transition rates. A perturbed experimental condition could affect both the energy of the states and the transition rates. Therefore, assuming the energy of a state can be updated in a constant time, the truncated CTMC can be updated in $O(|\hat{\mathcal{S}}| + |\hat{\mathcal{E}}|)$, where $\hat{\mathcal{E}}$ is set of transitions of the truncated CTMC. For nucleic acid kinetics the energy of a state can be computed from scratch in $O(b)$ time, or in $O(1)$ time using the energy calculations of a neighbor state which differs in one base pair (Schaeffer, 2013).

Quantifying the error. After we build truncated CTMCs, we need to quantify the error of MFPT estimates when experimental measurements are not available. It would help us set values for N, β, K and κ for fixed model parameters, and also evaluate when a truncated CTMC has a high error for perturbed model parameters. For exponential decay processes, one possible approach is to adapt the finite state projection FSP (Munsky and Khammash, 2006) method that is developed to quantify the error of truncated CTMCs for transient probabilities. We adapt it as follows. We combine all target states into one single absorbing state s_{f}. We project all states that are not in the truncated CTMC into an absorbing state s_o and we redirect all transitions from the truncated CTMC to states out of the CTMC into s_o. Then we use the standard matrix exponential equations to compute the full distribution on the state space at a given time. However, we only care about the probabilities that s_{f} and s_o are occupied. We search to compute the half-completion time $t_{1/2}$ with bounds by

$$t_{\min} \quad \text{s.t. } p(s_{\mathrm{f}}\,;\,t_{\min}) + p(s_o\,;\,t_{\min}) = \frac{1}{2}, \tag{5.6}$$

and

$$t_{\max} \quad \text{s.t. } p(s_{\mathrm{f}}\,;\,t_{\max}) = \frac{1}{2}, \tag{5.7}$$

where $p(s\,;\,t)$ is the probability that the process will be at state s at time t starting

from the set of initial states. Since s_{f} and s_o are the only absorbing states, then there exists a solution to Eq. 5.6 and clearly $t_{\min} \leq t_{1/2}$. Based on FSP, $p(s_{\mathrm{f}}\,;\, t_{\max})$ is an underestimate of the actual probability at time $t_{\max}$, if it exists. A possible way to determine if a solutions exists is to determine the probability of reaching state s_{f} compared to state s_o from the initial states, which can be calculated by solving a system of linear equations (see Eq. 2.13 from Metzner et al. (Metzner et al., 2009)). If the probability is greater or equal to $\frac{1}{2}$ then a solutions exists. If a solution does not exist for the given statespace, then based on FSP the error is guaranteed to decrease by adding more states and we can eventually find a solution to Eq. 5.7. The search for $t_{\max}$ can be completed with binary search. Thus, the true $t_{1/2}$ is guaranteed to satisfy $t_{\min} \leq t_{1/2} \leq t_{\max}$. For exponential decay processes, the relation between the half-completion time and the MFPT is (Cohen-Tannoudji et al., 1977; Simmons, 1972)

$$t_{1/2} = \frac{\ln 2}{\lambda} \text{ and } \tau = \frac{1}{\lambda} \rightarrow \tau = \frac{t_{1/2}}{\ln 2}, \tag{5.8}$$

where λ is the rate of the process. Thus, $\frac{t_{\min}}{\ln 2} \leq \tau \leq \frac{t_{\max}}{\ln 2}$.

A drawback of this approach is that we might need a large number of states to find a solution to Eq. 5.7, which might make the master equation or the linear system solver infeasible in practice. Efficiently quantifying the error of MFPT estimates in truncated CTMCs for exponential and non-exponential decay processes is beyond the scope of this work. It might also be possible to use some other existing work to evaluate a truncated state space (Backenköhler et al., 2019; Kuntz et al., 2019; Metzner et al., 2009).

In Section 5.4, we discuss some practical approaches to tune the parameters of pathway elaboration.

5.3 Experiments

Here, we conduct computational experiments to evaluate the pathway elaboration method. We implement pathway elaboration on top of the Multistrand kinetic simulator. Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/PathwayElaboration.

5.3.1 Dataset

We curate a dataset of 267 experimentally determined reaction rate constants from the published literature for hairpin closing, hairpin opening, helix association, helix dissociation with and without mismatches, and toehold-mediated three-way strand displacement, (Bonnet et al., 1998; Cisse et al., 2012; Hata et al., 2018; Machinek et al., 2014; Zhang et al., 2018). The dataset covers a wide range of slow and fast unimolecular and bimolecular reactions where the reaction rate constants vary over 8.6 orders of magnitude. The reactions are annotated with the temperature and the buffer conditions. Table 5.1 shows a summary of our dataset.

In datasets No. 1-6 from Table 5.1, we consider reactions that are feasible with SSA with our parameterization of Multistrand, given two weeks computation time, since we compare SSA results with pathway elaboration results. We indicate these reactions as $\mathcal{D}_{\text{train}}$ since we also use them as training set in our parameter estimation experiment. The reactions in datasets No. 7-8 are not feasible with SSA within two weeks. We indicate these reactions as $\mathcal{D}_{\text{test}}$ since we use them as testing set in our parameter estimation experiment.

To set π_0 for unimolecular reactions, we use particular complex microstates. For a bimolecular reaction, when the bimolecular transitions are slow enough between the two complexes, it is valid to assume the complexes each reach equilibrium before bimolecular transitions occur and therefore are Boltzmann distributed (Schaeffer, 2013). Given a volume and complex B therein, let $\mathcal{CM}$ be the set of all possible complex microstates of B. A distribution π_b is Boltzmann distributed with respect to complex B if and only if

$$\pi_b(c') = \frac{e^{-\Delta G(c')/RT}}{\sum_{c \in \mathcal{CM}} e^{-\Delta G(c)/RT}} \tag{5.9}$$

for all complex microstates $c' \in \mathcal{CM}$. In a bimolecular reaction of the form in Eq. 2.14, for a system microstate s that has complex microstates c and c' corresponding to complexes B and F, we define the initial distribution as $\pi_0(s) = \pi_b(c) \times \pi_b(c')$. For all other states, we define $\pi_0(s) = 0$.

For reactions in which we define only one target state, in the pathway construction step, we bias the paths towards that state. For reactions in which we define a

Table 5.1: Summary of the dataset of 267 experimentally determined reaction rate constants. The initial concentration of the reactants is denoted as u and k is the experimental reaction rate constant.

	Dataset No.	Reaction type & source	# of reactions	Mean # of bases	$[Na^+]$ (M)	T ($^\circ C$)	u (M)	$\log_{10} k$
$\mathcal{D}_{\text{train}}$	1	Hairpin opening (Bonnet et al., 1998)	63	25	0.1–0.5	10–49	1×10^{-8}	1.41–4.55
	2	Hairpin closing (Bonnet et al., 1998)	62	25	0.1–0.5	10–49	1×10^{-8}	3.36–4.76
	3	Helix dissociation w/ mismatch (Cisse et al., 2012)	39	18	0.01–0.2	23–37	1×10^{-8}	-1.19–0.93
	4	Helix association (Hata et al., 2018)††	43	46	0.195	25	5×10^{-8}	4.01–6.68
	5	Helix association (Zhang et al., 2018)††	20	72	0.75	37–55	1×10^{-5}	4.43–7.41
	6	Toehold-mediated three-way strand displacement w/ mismatch (Machinek et al., 2014)††	10	102	0.05†	23	5×10^{-9}–1×10^{-8}	5.33–6.78
$\mathcal{D}_{\text{test}}$	7	Helix association (Hata et al., 2018)††	4	46	0.195	25	5×10^{-8}	4.01–4.98
	8	Toehold-mediated three-way strand displacement w/ mismatch (Machinek et al., 2014)††	26	100	0.05†	23	5×10^{-9}–1×10^{-8}	2.71–6.33

† The experiment was performed without Na^+ in the buffer. We compute the free energy as if 50 mM $[Na^+]$ is present.

†† A bimolecular reaction. For bimolecular reactions, k^r has units $M^{-1}s^{-1}$. For unimolecular reactions, k^r has units s^{-1}.

set of target states, in this work, we bias paths towards only one target state, so that $\pi'(s_b) = 1$ for one state and $\pi'(s) = 0$ for all other states. For a hairpin opening reaction, we define the initial state to be the system microstate in which a strand has fully formed a duplex and a loop. We define the target state to be the system microstate in which the strand has no base pairs. Hairpin closing is the reverse reaction, where a strand with no base pair forms a fully formed duplex and a loop. For a helix dissociation reaction, we specify the initial state to be the system microstate in which two strands have fully formed a helix. We define the set of target states to be the set of system microstates in which the strands have detached and there are no base pairs within one of the strands. We bias paths towards the target state in which there are no base pairs formed within any of the strands. Helix association is the reverse bimolecular reaction, but we Boltzmann sample the initial reacting complexes in which the strands have not formed base pairs with each other. We define the target state to be the microstate in which the duplex has fully formed. In a three-way strand displacement, an invader strand displaces an incumbent strand in a duplex, where a toehold domain facilitates the reaction. We Boltzmann sample initial reacting complexes in which the incumbent and substrate form a complex through base pairing and the invader forms another complex. We define the set of target states to be the set of microstates where the incumbent is detached from the substrate and there are no base pairs within the incumbent. We bias paths towards the target state in which the substrate and invader have fully formed base pairs and there are no base pairs within the incumbent.

5.3.2 Experimental Setup

Experiments are performed on a system with 64 2.13GHz Intel Xeon processors and 128GB RAM in total, running openSUSE Leap 15.1. An experiment for a reaction is conducted on one processor. Our framework is implemented in Python, on top of the Multistrand kinetic simulator (Schaeffer, 2013; Schaeffer et al., 2015). To solve the matrix equations for estimating MFPTs in truncated CTMCs, we use the sparse direct solver from SciPy (Virtanen et al., 2020) when possible[1]. Otherwise we use the sparse iterative biconjugate gradient algorithm (Fletcher, 1976)

[1]The implementation we used allowed the sparse direct solver to use only up to 2GB of RAM

from SciPy.

In all of our experiments, the thermodynamic parameters for predicting the energy of the states are fixed and the energies are calculated with Multistrand. Each reaction uses its own experimental condition as provided in the dataset. For all experiments except for the Parameter Estimation section, we fix the parameters of the Multistrand kinetic model to the Metropolis Mode parameter set, that is $\theta_1 = \{k_{\text{uni}} \approx 2.41 \times 10^6\ \text{s}^{-1}, k_{\text{bi}} \approx 8.01 \times 10^5\ \text{M}^{-1}\text{s}^{-1}\}$. This parameter set has been obtained by calibrating the model on various types of reactions using semi-automatic truncated CTMCs (Zolaktaf et al., 2017). To obtain MFPTs with SSA, we use 1000 samples, except for three-way strand displacement reactions in which we use 100 samples, since the simulations take a longer time to complete.

5.3.3 Case Study

Here we illustrate the use of pathway elaboration, to gain insight on the kinetics of two contrasting toehold-mediated strand displacement reactions from Machinek et al. (2014).

Figures 5.3a and 5.4a show the two reactions that we consider (Machinek et al., 2014). In the reaction in Figure 5.3a, the invader and substrate are complementary strands in the displacement domain. In the reaction in Figure 5.4a, there is a mismatch between the invader and the substrate in the displacement domain. The rate of toehold-mediated strand displacement is usually determined by the time to complete the first bimolecular transition, in which the invader forms a base pair with the substrate for the first time. However, the rate could be controlled by several orders of magnitude by altering positions across the strand, such as using mismatch bases (Machinek et al., 2014). The reaction in Figure 5.4a is approximately 3 orders of magnitude slower than the reaction in Figure 5.3a. For the reaction in Figure 5.3a, $\log_{10} k = 6.43$, $\log_{10} \hat{k}_{\text{PE}} = 6.62$, $\log_{10} \hat{k}_{\text{SSA}} = 6.75$, $|\hat{\mathcal{S}}| = 4.3 \times 10^5$, the computation time of pathway elaboration is 1.4×10^5 s, and the computation time of SSA is 3.9×10^5 s. For the reaction in Figure 5.4a, $\log_{10} k = 3.17$, $\log_{10} \hat{k}_{\text{PE}} = 3.59$, $|\hat{\mathcal{S}}| = 7 \times 10^5$, the computation time of pathway elaboration is 2.7×10^5 s, and SSA is not feasible within 1×10^6 s.

In Figures 5.3b-5.3d and Figures 5.4b-5.4d, we illustrate different properties

of the truncated CTMCs for the reactions in Figures 5.3a and 5.4a, respectively. Comparing Figure 5.3b with Figure 5.4b, we see that many states are sampled midway in Figure 5.4b due to the mismatch. In Figures 5.3c and 5.4c, we compare the energy barrier (increase in free energy) while moving from the beginning of the x-axis towards the end of the x-axis. In Figure 5.3c, we can see a noticeable energy barrier in the beginning. However, in Figure 5.4c, we can see two noticeable energy barriers, one in the beginning and one midway. Figures 5.3d and 5.4d show states that are δ-close to the target states. These figures show that with δ-pruning, states that are further from the initial states and closer to the target states will be pruned with smaller values of δ, compared to states that are closer to the initial states and further from the target states. Comparing Figure 5.3d with Figure 5.4d, the states quickly reach the target states after the first several transitions in Figure 5.3d (after the energy barrier). However, in Figure 5.4d, the states do not quickly reach the target states until after the second energy barrier. Figure 5.3e and 5.4e show the free energy landscape and some of the secondary structures for a random path from an initial state to a target state for the reactions in Figures 5.3a and 5.4a, respectively. For the reaction in Figure 5.3a, the barrier is near the first transition. For the reaction in Figure 5.4a, there is a noticeable barrier after several base pairs form between the invader and the substrate, presumably near the mismatch.

5.3.4 Mean First Passage Time and Reaction Rate Constant Estimation

To evaluate the estimations of pathway elaboration, we compare its estimations with estimations obtained from SSA for the reactions in $\mathcal{D}_{\text{train}}$. Note that for many of these reactions the size of the state space is exponentially large in the length of the strands. Therefore, exact matrix equations is not possible for them. Instead we use SSA since it can generate statistically correct trajectories. We also compare the wall-clock computation time of pathway elaboration with SSA for these reactions.

We evaluate the estimations of pathway elaboration based on the mean absolute error (MAE) with SSA. We define the MAE of pathway elaboration with SSA over

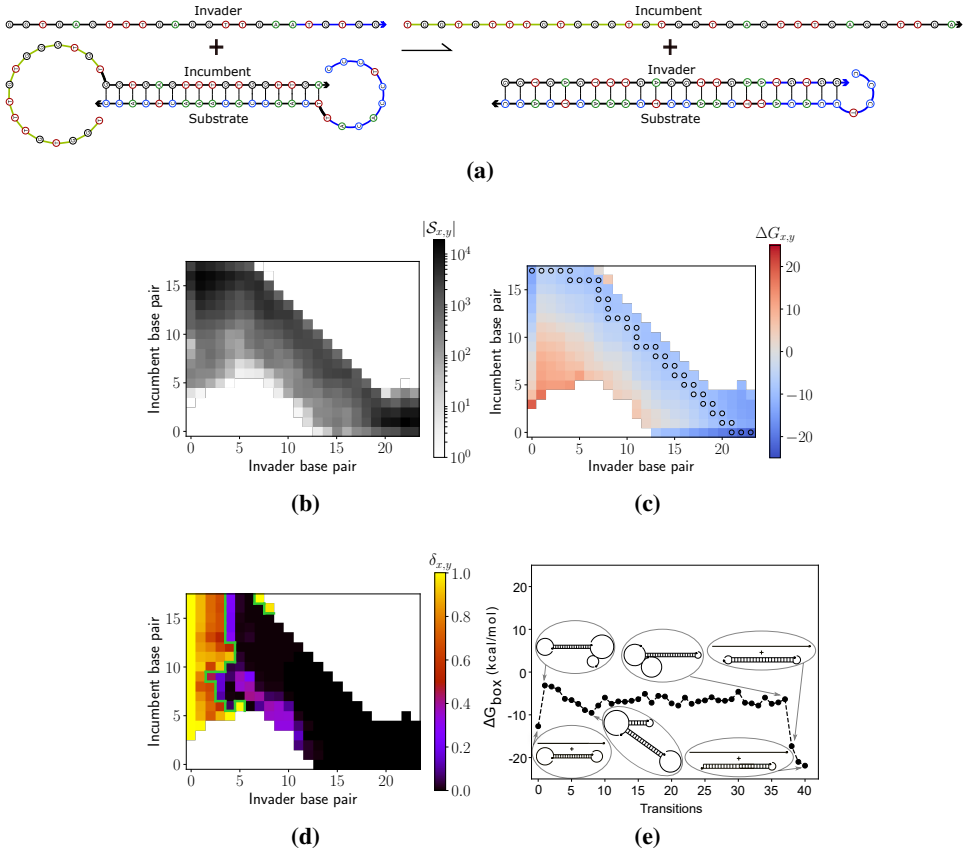


Figure 5.3: Results of truncated CTMCs built with pathway elaboration ($N = 128$, $\beta = 0.6$, $K = 1024$ and $\kappa = 16$ ns) for (**a**) a toehold-mediated three-way strand displacement reaction that has a 6-nt toehold and a 17-nt displacement domain (Machinek et al., 2014). In Figures 5.3b, 5.3c, and 5.3d, the x-axis corresponds to the number of base pairs between the invader and the substrate, and the y-axis corresponds to the number of base pairs between the incumbent and the substrate. (**b**) At coordinate (x,y), $|\mathcal{S}_{x,y}|$ is shown, where $\mathcal{S}_{x,y}$ is a system macrostate equal to the set of states with coordinate (x,y). (**c**) At coordinate (x,y), the free energy $\Delta G_{x,y}$ is shown, which is defined as $\Delta G_{x,y} = -RT \ln \sum_{s \in \mathcal{S}_{x,y}} e^{\frac{-\Delta G(s)}{RT}}$ (Schaeffer, 2013). The free energy of the path in Figure 5.3e is also shown with the $\circ$ marker in Figure 5.3c. (**d**) At coordinate (x,y), the value of $\delta_{x,y} = \sum_{s \in \mathcal{S}_{x,y}} \frac{w_s \delta(s)}{\sum_{s \in \mathcal{S}_{x,y}} w_s}$ is shown, where $\delta(s) = \mathbb{E}[\tau_s]/\tau_{\pi_0}$ and $w_s = e^{\frac{-\Delta G(s)}{RT}}$. For ease of understanding, the green "halfway line" separates coordinates where $\delta_{x,y}$ is greater than 0.5 from coordinates where $\delta_{x,y}$ is less than 0.5. (**e**) The free energy landscape of a random path built with pathway elaboration ($N = 1$, $\beta = 0$, $K = 0$ and $\kappa = 0$ ns) and the initial and the final states and some states near the local extrema are illustrated.

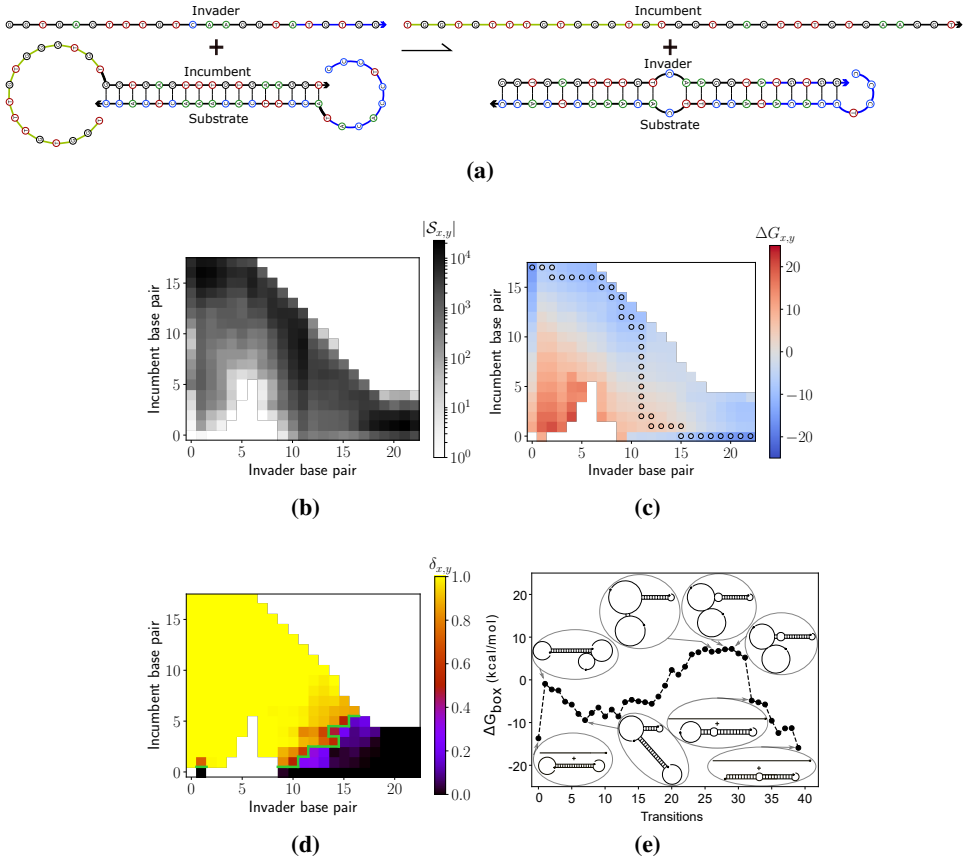


Figure 5.4: As in Figure 5.3, results of truncated CTMCs built with pathway elaboration ($N = 128$, $\beta = 0.6$, $K = 1024$ and $\kappa = 16$ ns) for a toehold-mediated three-way strand displacement reaction that has a 6-nt toehold, a 17-nt displacement domain, and a mismatch exists between the invader and the substrate at position 6 of the displacement domain (Machinek et al., 2014).

a dataset $\mathcal{D}$ as

$$\begin{aligned} \text{MAE} = \frac{1}{|\mathcal{D}|} \sum_{r \in \mathcal{D}} |\log_{10} \hat{\tau}^r_{\text{SSA}} - \log_{10} \hat{\tau}^r_{\text{PE}}| = \\ \frac{1}{|\mathcal{D}|} \sum_{r \in \mathcal{D}} |\log_{10} \hat{k}^r_{\text{SSA}} - \log_{10} \hat{k}^r_{\text{PE}}|, \end{aligned} \tag{5.10}$$

where $\hat{\tau}^r_{\text{PE}}$ and $\hat{\tau}^r_{\text{SSA}}$ are the estimated MFPTs of SSA and pathway elaboration for

reaction r, respectively, and $\hat{k}^r_{\text{SSA}}$ and $\hat{k}^r_{\text{PE}}$ are the estimated reaction rate constants of SSA and pathway elaboration for reaction r, respectively. The equality follows from Eqs. 2.15 and 2.16. We use $\log_{10}$ differences since the reactions rate constants cover many orders of magnitude. Note that we could use other metrics instead of the MAE to compare pathway elaboration with SSA. However, the MAE is conceptually easy to understand and since we are using the MAE of the $\log_{10}$ values, we can understand on average how off the results are. For example, an MAE of 1 means on average the predictions are off by a factor of 10. In the rest of this section, we first look at the trade-off between the MAEs and the size of the truncated state space set $\hat{\mathcal{S}}$, with regards to different parameter settings of the pathway elaboration method. Then we look at the trade-off between the MAE and the computation time.

MAE versus $|\hat{\mathcal{S}}|$. Figure 5.5 shows the MAE versus $|\hat{\mathcal{S}}|$ of pathway elaboration for different configurations of the N, β, K, and κ parameters. Figure B1 and B2 from the Appendix represent Figure 5.5 by varying only two parameters at a time. The figures show that generally as N and β increase, the MAE decreases. This is because for a fixed N as $\beta \to 1$ the ensemble of paths will be generated by SSA. As $N \to \infty$, the truncated state space becomes larger and is more likely to contain the most probable paths from the initial states to the target states.

Comparing the MAE of configurations where $K = 0$ and $\kappa = 0$ with other settings where $K > 0$ and $\kappa > 0$, shows that the elaboration step helps reduce the MAE (in the Appendix, compare Figures B1a-B1d with Figures B1i-B1l). Particularly, the elaboration step is useful for dataset No. 4, helix association from Zhang et al. (2018) where intra-strand base pairs can form before completing hybridization. The plots show that the elaboration step is more useful when β is small (in the Appendix, compare Figures B2a-B2d with Figures B2i-B2l). This could be because elaboration helps find rate determining states that were not explored due to the biased sampling. When $\beta \to 1$ the pathway elaboration method will perform as SSA and rate determining states can be found without elaboration.

Furthermore, the figures show that as K increases, the MAE decreases. However, with a large value for κ and a small value of K the performance could be diminished (such as in Figure B2c of the Appendix). In particular, consider that the K and κ might involve simulations that go on excursions outside the "main"

densely-visited parts of the enumerated state space, and they might even terminate out there. Such excursions might very well introduce significant local minima into the enumerated state space - even when no significant local minima exist in the original full state space. For example, consider an excursion that goes off-path down a wide slope, perhaps toward the target state. If it terminates before reaching a target state, then a hypothetical simulation in the enumerated state space could get stuck, needing to climb back up the slope to the point where the excursion began. The expected hitting time in the enumerated state space will account for such wasted time, thus leading to an over estimation of the MFPT. Therefore, κ should be tuned with respect to K.

MAE versus computation time. Table 5.2 illustrates the MAE and computation time of pathway elaboration for when $N = 128$, $\beta = 0.4$, $K = 256$, and $\kappa = 16$ ns compared with SSA. We use this parameter setting here because it provides a good trade-off between accuracy and computational time for the larger reactions. For the smaller reactions, we could achieve the same MAE with less computational time (by using smaller values for the parameter setting). Figure 5.6 further shows the prediction of pathway elaboration for this parameter setting compared to the prediction of SSA for individual reactions. In Table 5.2, the MAE for unimolecular reactions is smaller than 0.05, whereas for bimolecular reactions it is larger than 0.29. This is because the CTMCs for the bimolecular reactions in our dataset are naturally bigger than the CTMCs for the unimolecular reactions in our dataset, and require larger truncated CTMCs. The MAE can be further reduced by changing the parameters (as shown in Figure 5.5). With our implementation of pathway elaboration, the computation time of pathway elaboration for datasets No. 3, No. 4, and No. 6 are 2 times, 20 times, 3 times smaller than SSA, respectively. The computation time of SSA for datasets No. 1, No. 2, and No. 5 is smaller than the computation time of pathway elaboration. This is because pathway elaboration has some overhead, and in cases where SSA is already fast it can be slow. However, as we later show in the experiments, even for these type of reactions, pathway elaboration could still be useful for building truncated CTMCs for the rapid evaluation of perturbed parameters. The computation time for pathway elaboration could be significantly improved with more efficient implementations of the method.

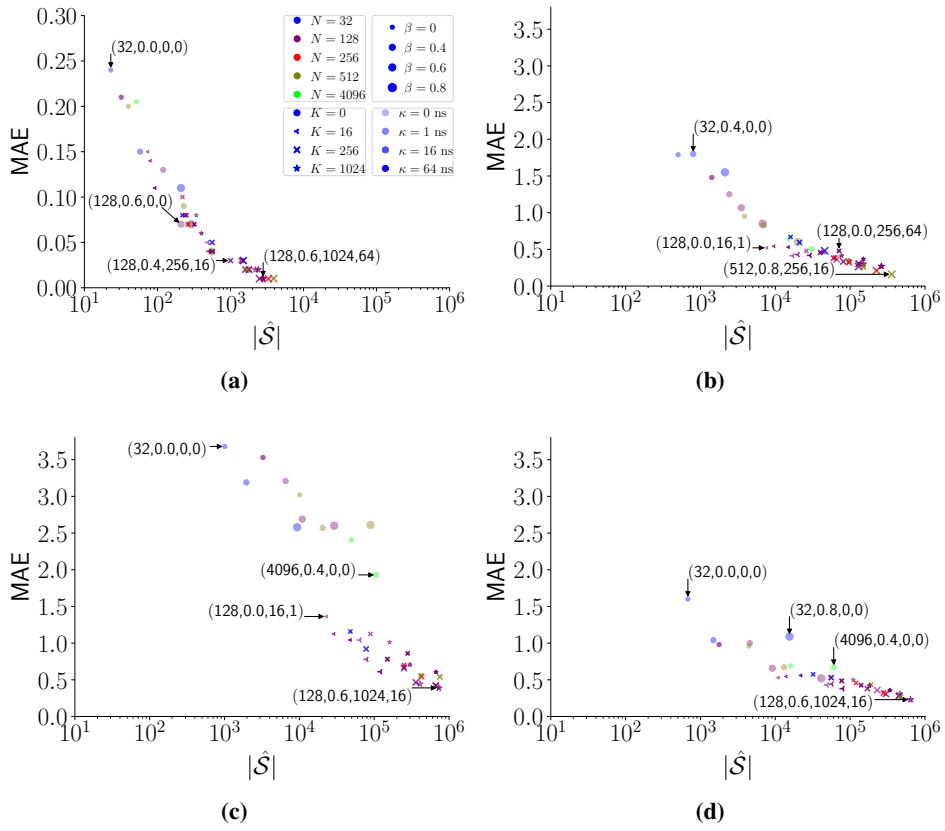


Figure 5.5: MAE vs $|\hat{\mathcal{S}}|$ for different values of N, β, K and κ. (**a**) datasets No. 1,2, and 3, (**b**) dataset No. 4, (**c**) dataset No. 5, and (**d**) dataset No. 6. The configuration of N, β, K, and κ corresponding to the truncated CTMCs which have the minimum and maximum MAE are shown on each plot.

5.3.5 δ-Pruning

Figure 5.7 shows how δ-pruning affects the quality of the $\log_{10}$ reaction rate constant estimates, the size of the state spaces, and the computation time of solving the matrix equations, for dataset No. 6. The MFPT estimates satisfy the bound given by Eq. 5.5 whilst δ-pruning reduces the computation time for solving the matrix equations by an order of magnitude for $\delta = 0.6$. Using larger values of δ we can further decrease the computation time. If we reuse the CTMCs many times, such as in parameter estimation, δ-pruning could help reduce computation

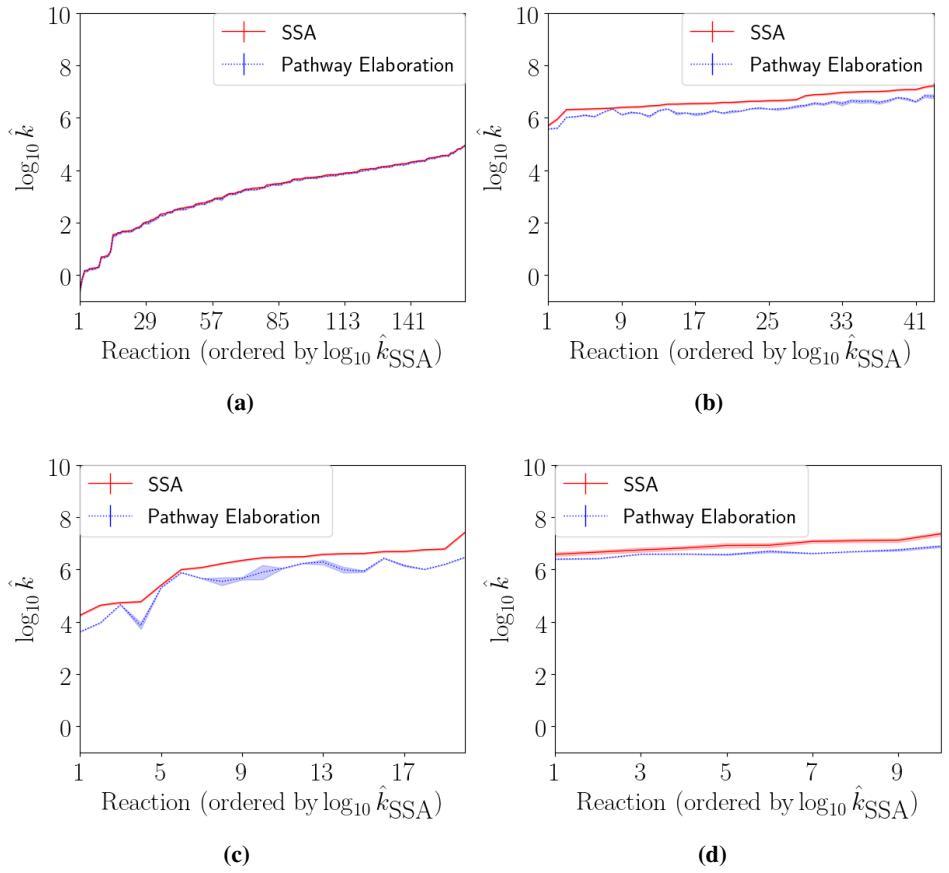


Figure 5.6: The $\log_{10}\hat{k}_{\mathrm{SSA}}$ and $\log_{10}\hat{k}_{\mathrm{PE}}$ ($N = 128$, $\beta = 0.6$, $K = 256$, and $\kappa =$ 16 ns) for (**a**) datasets No. 1,2, and 3, and (**b**) dataset No. 4, (**c**) dataset No. 5, and (**d**) dataset No. 6. The reactions are ordered along the x-axis by their predicted $\log_{10}\hat{k}_{\mathrm{SSA}}$. The pathway elaboration experiments are repeated three times. For each reaction, $\log_{10}\hat{k}_{\mathrm{PE}}$ is calculated by the average of the three experiments. The error bars for pathway elaboration indicate the range (minimum to maximum) of the three experiments. The error bars for SSA indicate the 95% percentile bootstrap of the $\log_{10}\hat{k}_{\mathrm{SSA}}$.

Table 5.2: The statistics of pathway elaboration ($N = 128$, $\beta = 0.6$, $K = 256$, and $\kappa = 16$ ns) versus SSA. All statistics are averaged over the '# of reactions'. MAE refers to the Mean Average Error of pathway elaboration with SSA (see Eq. 5.10). $|\hat{\mathcal{S}}|$ is the size of the truncated state space set.

Dataset No.	Reaction type & source	# of reactions	MAE	Mean $\lvert\hat{\mathcal{S}}\rvert$ for pathway elaboration	Mean matrix equation computation time (s) for pathway elaboration	Mean computation time (s) for pathway elaboration	Mean computation time (s) for SSA
1	Hairpin opening (Bonnet et al., 1998)	63	0.04	5.7×10^2	4.5×10^{-3}	1.0×10^3	2.7×10^1
2	Hairpin closing (Bonnet et al., 1998)	62	0.03	1.8×10^3	1.5×10^{-2}	1.0×10^3	1.2×10^1
3	Helix dissociation w/ mismatch (Cisse et al., 2012)	39	0.04	5.3×10^2	6.8×10^{-3}	1.6×10^3	3.8×10^3
4	Helix association (Hata et al., 2018)	43	0.29	8.1×10^4	3.0×10^1	2.1×10^4	4.9×10^5
5	Helix association (Zhang et al., 2018)	20	0.51	3.8×10^5	2.3×10^4	1.6×10^5	3.7×10^4
6	Three-way strand displacement w/ mismatch (Machinek et al., 2014)	10	0.31	3.0×10^5	1.3×10^3	1.3×10^5	3.8×10^5
	All reactions	237	0.13	6.0×10^4	2.0×10^3	2.4×10^4	1.1×10^5

time significantly.

5.3.6 Parameter Estimation

In the previous subsections, the underlying parameters of the CTMCs were fixed. In this subsection, we assume the parameters of the kinetic model of the CTMCs are not calibrated and we use pathway elaboration to build truncated CTMCs that we reuse to rapidly evaluate perturbed parameter sets during parameter estimation.

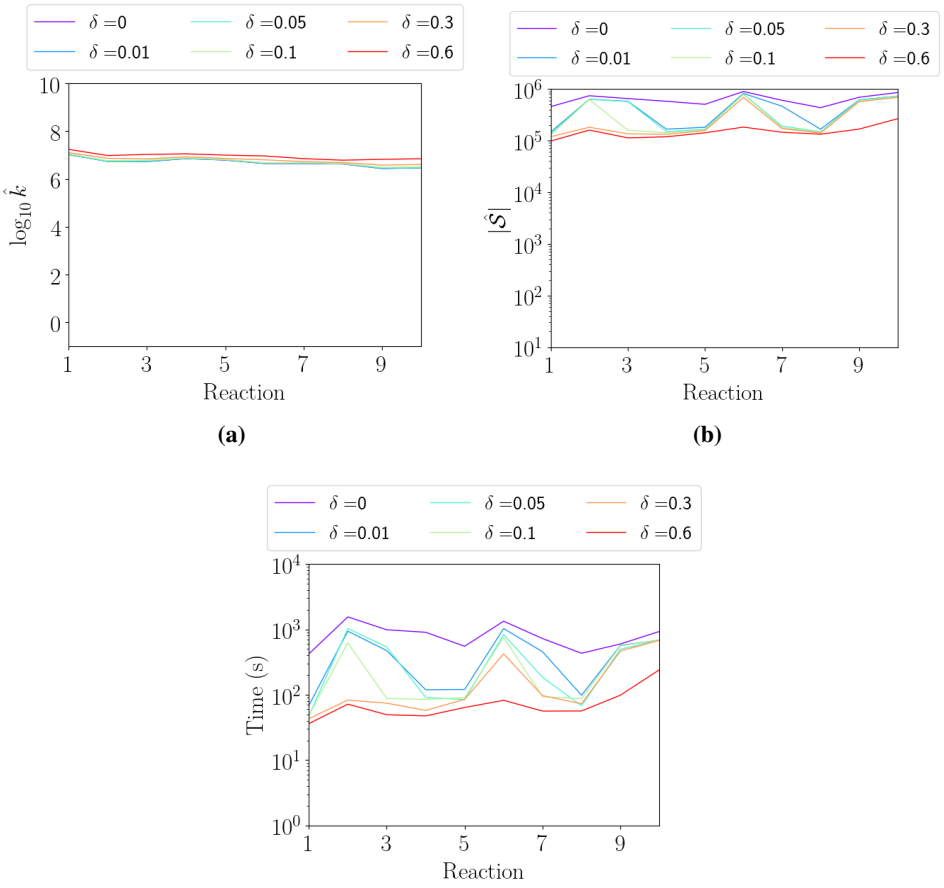


Figure 5.7: The effect of δ-pruning with different values on truncated CTMCs that are built with the pathway elaboration method ($N = 128$, $\beta = 0.6$, $K = 1024$, $\kappa = 16$ ns) for dataset No. 6. $\delta = 0$ indicates δ-pruning is not used. (**a**) The $\log_{10}\hat{k}$. (**b**) The size of the truncated state space $|\hat{\mathcal{S}}|$. (**c**) The computation time for solving the matrix equation in Eq. 2.7.

We use the 237 reactions indicated as $\mathcal{D}_{\text{train}}$ in Table 5.1 to optimize an initial parameter set. We use the 30 rare event reactions indicated as $\mathcal{D}_{\text{test}}$ in Table 5.1 to show that given a calibrated parameter set of the CTMC model, the pathway elaboration method can estimate MFPTs and reaction rate constants of reactions close to their experimental measurement.

To optimize the parameter set, we use a similar approach to the fixed pathway ensemble inference (FPEI) method (Zolaktaf et al., 2019). We seek the parameter set that minimizes the mean squared error (MSE) as

$$\begin{aligned}\theta^* = \underset{\theta}{\text{argmin}}\ \frac{1}{|\mathcal{D}_{\text{train}}|}\sum_{r\in\mathcal{D}_{\text{train}}}\left(\log_{10}\tau^r - \log_{10}\hat{\tau}^r_{\text{PE}}(\theta)\right)^2 = \\ \underset{\theta}{\text{argmin}}\ \frac{1}{|\mathcal{D}_{\text{train}}|}\sum_{r\in\mathcal{D}_{\text{train}}}\left(\log_{10}k^r - \log_{10}\hat{k}^r_{\text{PE}}(\theta)\right)^2,\end{aligned} \tag{5.11}$$

which is a common cost function for regression problems. We use the Nelder-Mead optimization algorithm (Nelder and Mead, 1965; Virtanen et al., 2020) to minimize the MSE. The equality follows from Eqs. 2.15 and 2.16. We initialize the simplex in the algorithm with $\theta_2 = \{k_{\text{uni}} = 5\times10^4\ \text{s}^{-1}, k_{\text{bi}} = 5\times10^4\ \text{M}^{-1}\text{s}^{-1}\}$ in which we choose arbitrarily and two perturbed parameter sets. Each perturbed parameter set is obtained from θ_2 by multiplying one of the parameters by 1.05, which is the default implementation of the optimization software. For every reaction, we also initialize the Multistrand kinetic model with θ_2. We build truncated CTMCs with pathway elaboration ($N = 128$, $\beta = 0.4$, $K = 256$, $\kappa = 16$ ns). Whenever the matrix equation solving time is large (here we consider a time of 120 s large), we use δ-pruning (here we use δ values of $0.01 - 0.6$) to reduce the time. During the optimization, for a new parameter set we update the parameters in the kinetic model of the truncated CTMCs and we reuse the truncated CTMC to evaluate the parameters. Similar to FPEI, to reduce the bias and to ensure that the truncated CTMCs are fair with respect to the optimized parameters, we can occasionally rebuild truncated CTMCs from scratch.

Although we use the MSE of pathway elaboration with experimental measurements as our cost function in the optimization procedure, the MAE of pathway elaboration with experimental measurements also decreases. Figure 5.8 shows how the parameters, the MSE, and the MAE change during optimization. The markers

are annotated with the MSE and the MAE of $\mathcal{D}_{\text{train}}$, dataset No. 7, and dataset No. 8 when truncated CTMCs are built from scratch. The MAE of $\mathcal{D}_{\text{train}}$ with the initial parameter set θ_2 is 1.43, but the algorithm finds $\theta^* = \{k_{\text{uni}} \approx 3.61 \times 10^6 \text{ s}^{-1}, k_{\text{bi}} \approx 1.12 \times 10^5 \text{ M}^{-1} \text{ s}^{-1}\}$ and reduces the MAE of $\mathcal{D}_{\text{train}}$ to 0.46. The MAE of dataset No. 7 and dataset No. 8, which are not used in the optimization, reduce from 2.00 to 0.73 and from 1.00 to 0.63, respectively.

For a fixed setting of the pathway elaboration method parameters and the parameter set of the CTMC model, given an estimate of the MAE of pathway elaboration with SSA ($\text{MAE}^{\mathcal{D}}_{\text{PE,SSA}}$) and an estimate of the MAE of pathway elaboration with experimental measurements ($\text{MAE}^{\mathcal{D}}_{\text{PE,Experiments}}$), we can obtain an upper bound on the MAE of SSA with experimental measurements ($\text{MAE}^{\mathcal{D}}_{\text{SSA,Experiments}}$) as

$$\text{MAE}^{\mathcal{D}}_{\text{SSA,Experiments}} \leq \text{MAE}^{\mathcal{D}}_{\text{PE,SSA}} + \text{MAE}^{\mathcal{D}}_{\text{PE,Experiments}}, \tag{5.12}$$

which follows from the triangle inequality of the $L1$ norm. Given an upper bound on $\text{MAE}^{\mathcal{D}}_{\text{PE,SSA}}$ that does not depend on the parameter set of the CTMC model, we can tighten this inequality by decreasing $\text{MAE}^{\mathcal{D}}_{\text{PE,Experiments}}$. Although we do not currently have an analytical upper bound on $\text{MAE}^{\mathcal{D}}_{\text{PE,SSA}}$, in Table 5.2 and Figures 5.5 and 5.6, we numerically showed that we can achieve a reasonable value for $\text{MAE}^{\mathcal{D}}_{\text{PE,SSA}}$ with a good parameter setting for pathway elaboration.

Overall, the experiment in this subsection shows that pathway elaboration enables MFPT estimation of rare events. It predicts their MFPTs close to their experimental measurements given an accurately calibrated model for their CTMCs. Moreover, it shows that pathway elaboration enables the rapid evaluation of perturbed parameters and makes feasible tasks such as parameter estimation which benefit from such methods. On average for the 30 reactions in the testing set, pathway elaboration takes less than two days, whereas SSA is not feasible within two weeks. The entire experiment in Figure 6 takes less than five days parallelized on 40 processors. Note that clearly our optimization procedure could be improved, for example by using a larger dataset or a more flexible kinetic model (Zolaktaf et al., 2017). However, the experiment in this subsection is only a preliminary study; we leave a rigorous study on calibrating nucleic acid kinetic models with pathway elaboration and possible improvements to future studies.

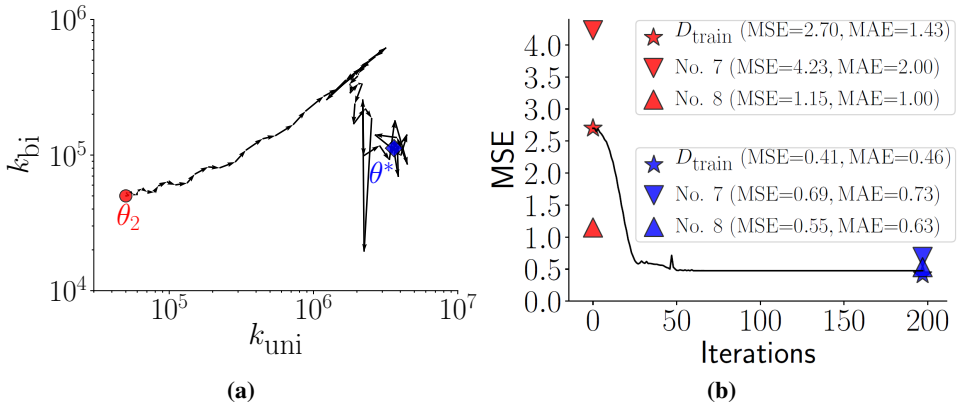


Figure 5.8: Results of parameter estimation using pathway elaboration ($N = 128$, $\beta = 0.4$, $K = 256$, $\kappa = 16$ ns). The optimization uses 237 reactions from $\mathcal{D}_{\text{train}}$. (**a**) The parameters are optimized from an initial simplex of θ_2 and its perturbations to $\theta^* = \{k_{\text{uni}} \approx 3.61 \times 10^6\ \text{s}^{-1}, k_{\text{bi}} \approx 1.12 \times 10^5\ \text{M}^{-1}\ \text{s}^{-1}\}$. (**b**) The parameters are optimized using $\mathcal{D}_{\text{train}}$, shown with a line graph, and evaluated on dataset No. 7 and No. 8. The markers are annotated with the MSE and MAE of $\mathcal{D}_{\text{train}}$, dataset No. 7, and dataset No. 8 when the truncated CTMCs are built from scratch and parameterized with θ_2 and θ^*.

5.4 Summary and Directions for Future Improvements

In this chapter, we address the problem of estimating MFPTs of rare events in large CTMCs, and also the rapid evaluation of perturbed parameters. We propose the pathway elaboration method for detailed-balance CTMCs, which is a time-efficient probabilistic truncation-based approach for MFPT estimation. We conduct computational experiments on a wide range of experimental measurements to show pathway elaboration is suitable for estimating the rates of nucleic acid rare events. In summary, our results are promising, but there is still room for improvement.

Using pathway elaboration, in the best possible case, the sampled region of states and transitions is obtained faster than SSA, but without significant bias in the collected states and transitions. The sampled region may however qualitatively differ from what would be obtained from SSA, which may compromise the MFPTs estimated via this method. Moreover, reusing truncated CTMCs for significantly perturbed parameters could lead to inaccurate estimation of the MFPT of the origi-

nal CTMC. In Section 5.2, for exponential decay processes, we introduce a method that could help us quantify the error of the MFPT estimate. However, since we have to search for the bounds, it might be slow in practice. So how can we efficiently tune these parameters in practice? Similar to SSA, for a fixed β and when $K = 0$ and $\kappa = 0$, we could increase N until the estimated MFPT stops changing significantly (based on the law of large numbers it will converge). Note that for $K = 0$ and $\kappa = 0$ we could compute the MFPT by computing the average of the biased paths without solving matrix equations. As shown in Proposition 1, if we set β to less than $1/2$, then biased paths will reach target states in expected time that is linear in the distance from initial to target states. For setting K, one possibility is to consider the number of neighbors of each state. A reaction where states have a lot of neighbors requires a larger K compared to a reaction where states have a smaller number of neighbors. κ should be set with respect to K. As stated in Section 5.3, a large value of κ along with a small value of K could result in excursions that do not reach any target state and lead to overestimates of the MFPT. One could set κ to a small value and then increase K until the MFPT estimate stops changing, and could repeat this process while feasible.

In the pathway elaboration method, we estimate MFPTs by solving matrix equations. Thus, its performance depends on the accuracy and speed of matrix equation solvers. For example, applying matrix equation solvers may not be suitable if the initial states lie very far from the target states, since the size of the truncated CTMCs depends on the shortest-path distance between these states. Although solving matrix equations through direct and iterative methods has progressed, both theoretically and practically (Cohen et al., 2018; Fletcher, 1976; Parks et al., 2006; Virtanen et al., 2020), solving stiff (multiple time scales) or very large equations could still be problematic in practice. More stable and faster solvers would allow us to estimate MFPTs for stiffer and larger truncated CTMCs. Moreover, it might be possible to use fast updates for matrix decomposition algorithms (Brand, 2006; Bunch and Nielsen, 1978). So if we require to compute MFPT estimates with matrix equations as we monotonically grow the size of the state space, the total cost for solving all the linear systems would be the same cost as solving the final linear system from scratch.

We might be able to improve the pathway elaboration method to relieve the

limitations discussed above. For example, it might be possible to use an ensemble of truncated CTMCs to obtain an unbiased estimate of the MFPT (Georgoulas et al., 2017). To avoid excursions that lead to overestimation of the MFPT in the state elaboration step, we could run the pathway construction step from the last states visited in the state elaboration step. This would also relax the constraint of having reversible or detailed balance transitions. Presumably, an alternating approach of the two steps would make the approach more flexible. Moreover, currently we run the state elaboration step from every state of the pathway for a fixed setting. However, it might not be necessary to run the elaboration from all states with the same setting. Efficiently running the state elaboration step as necessary, could reduce the time to construct the truncated CTMC in addition to the matrix equation solving time.

Finally, we evaluated the pathway elaboration method for predicting the MFPT of interacting nucleic acid strands. However, the method is generally applicable to detailed-balance CTMC models. Thus, it would be useful to evaluate it for other applications, such as protein folding, chemical reaction networks, and molecular evolution.

Chapter 6

Summary

In this thesis, we introduced a new elementary step kinetic model of interacting nucleic acid strands. We also addressed MFPT estimation and the rapid evaluation of perturbed parameters in the full state space of reactions' CTMCs, in order to efficiently make kinetic estimations and to calibrate kinetic models.

In Chapter 3, we reported the initial results of our effort to develop accurate kinetic models for nucleic acids. We introduced the Arrhenius kinetic model. Our model is derived from the Metropolis model, but its transition rates depend on activation energy and on the immediate local environment surrounding the affected base pair. To calibrate and evaluate these models, we compiled a dataset of 376 experimentally determined reaction rate constants that we sourced from existing publications and cover a wide range of reactions, including hairpin closing, hairpin opening, bubble closing, helix association, helix dissociation, toehold-mediated three-way strand displacement, and toehold-mediated four-way strand exchange (Altan-Bonnet et al., 2003; Bonnet, 2000; Bonnet et al., 1998; Dabby, 2013; Kim et al., 2006; Machinek et al., 2014; Morrison and Stols, 1993; Reynaldo et al., 2000; Zhang and Winfree, 2009). We showed how to infer model parameters using an ensemble Markov chain Monte Carlo (MCMC) approach and a maximum a posteriori (MAP) approach. To evaluate the likelihood, we computed MFPTs and reaction rate constants using exact matrix computation on simplified state spaces for the reactions in our dataset. Overall, our results are encouraging and suggest that the new Arrhenius kinetic model, calibrated sufficiently, outper-

forms the Metropolis kinetic model. Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/ArrheniusInference.

In Chapter 4, we addressed MFPT estimation and the rapid evaluation of perturbed parameters for parameter inference in the full state space of reactions' CTMCs. We showed how to use a reduced variance stochastic simulation algorithm (RVSSA) to estimate MFPTs faster than SSA. We introduced the fixed path ensemble inference (FPEI) method to speed up parameter inference. We showed how to estimate model parameters in the full state of the reactions CTMCs using a generalized method of moments (GMM) estimator (Hansen, 1982). We conducted computational experiments on a dataset of 21 experimental DNA reactions that have moderate or large state spaces or are slow. The dataset consists of hairpin closing, hairpin opening, helix association, and helix dissociation with and without mismatches (Bonnet et al., 1998; Cisse et al., 2012; Hata et al., 2018; Wetmur, 1976). Overall, our results are encouraging and show that RVSSA may be useful for estimating MFPTs for some reactions and that FPEI speeds-up parameter inference, although the methods are currently not applicable to reactions that happens on a long time scale (rare events). Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/FPEI.

In Chapter 5, similar to Chapter 4, we addressed MFPT estimation and the rapid evaluation of perturbed parameters in the full state space of reactions' CTMCs. However, we proposed a method, called pathway elaboration, which is applicable to reactions that happen on a long time scale, that is rare events. We conducted computational experiments on a dataset of 267 reactions covering different type of reactions, namely, hairpin opening, hairpin closing, helix dissociation with and without mismatches, helix association, and toehold-mediated three-way strand displacement with and without mismatches (Bonnet et al., 1998; Cisse et al., 2012; Hata et al., 2018; Machinek et al., 2014; Zhang et al., 2018). The dataset covers a wide range of reaction rate constants and includes reactions that have more than 100 bases in their strands. Overall, our extensive computational experiments, including a case study, show that pathway elaboration may be useful for efficiently estimating the kinetics of nucleic acids that are modeled as CTMCs, including rare event reactions in large state spaces. Our framework and the dataset are available at https://github.com/DNA-and-Natural-Algorithms-Group/PathwayElaboration.

All in all, we believe this thesis could improve the prediction of kinetics for interacting nucleic acid strands modeled as CTMCs and could facilitate the design of nucleic acid-based devices. Although our results are promising, they could be improved as discussed at the end of each chapter. In brief, we need to comprehensively calibrate the Arrhenius kinetic model in the full state space of reactions. In this direction, the dataset needs to be expanded to complete the evaluation of kinetic models, including considering RNA strands. There are also some possibilities for further improvement of the FPEI and the pathway elaboration methods. After the Arrhenius model is sufficiently calibrated, similar to our case study in Chapter 5 and using the pathway elaboration method, it might be possible to predict and analyze the kinetics of some interacting nucleic acid strands that are used in nucleic acid-based devices, such as toehold switches (Green et al., 2014) and oscillators (Srinivas et al., 2017).

Finally, the FPEI and the pathway elaboration methods are not specific to nucleic acid kinetics and are generally applicable to other applications that are modeled as CTMCs, such as protein folding (McGibbon and Pande, 2015), chemical reaction networks (Anderson and Kurtz, 2011; Cappelletti et al., 2020; Soloveichik et al., 2008), and molecular evolution (Liò and Goldman, 1998). It would be useful to possibly adapt and evaluate these methods for such applications of CTMCs.

Bibliography

Aalberts, D. P., Parman, J. M., and Goddard, N. L. Single-strand stacking free energy from DNA beacon kinetics. *Biophysical Journal*, 84(5):3212–3217, 2003. → page 42

Allen, R. J., Frenkel, D., and ten Wolde, P. R. Simulating rare events in equilibrium or nonequilibrium stochastic systems. *The Journal of Chemical Physics*, 124(2):024102, 2006. → page 22

Allen, R. J., Valeriani, C., and ten Wolde, P. R. Forward flux sampling for rare event simulations. *Journal of Physics: Condensed Matter*, 21(46):463102, 2009. → pages 4, 22, 23

Altan-Bonnet, G., Libchaber, A., and Krichevsky, O. Bubble dynamics in double-stranded DNA. *Physical Review Letters*, 90(13):138101, 2003. → pages xiv, 15, 27, 29, 30, 93, 124

Amato, N. M. and Song, G. Using motion planning to study protein folding pathways. *Journal of Computational Biology*, 9(2):149–168, 2002. → page 25

Anderson, D. F. and Kurtz, T. G. Continuous time Markov chain models for chemical reaction networks. In *Design and Analysis of Biomolecular Circuits*, pages 3–42. Springer, 2011. → pages 3, 7, 95

Andrieu, C. and Roberts, G. O. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725, 2009. → page 24

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. An introduction to MCMC for machine learning. *Machine learning*, 50(1-2):5–43, 2003. → pages 22, 67

Andronescu, M., Aguirre-Hernandez, R., Condon, A., and Hoos, H. H. RNAsoft: a suite of RNA secondary structure prediction and design software tools. *Nucleic Acids Research*, 31(13):3416–3422, 2003. → pages 3, 4, 21, 26

Andronescu, M., Condon, A., Hoos, H. H., Mathews, D. H., and Murphy, K. P. Computational approaches for RNA energy parameter estimation. *RNA*, 16(12): 2304–2318, 2010. → pages 4, 20, 26

Angenent-Mari, N. M., Garruss, A. S., Soenksen, L. R., Church, G., and Collins, J. J. A deep learning approach to programmable rna switches. *Nature communications*, 11(1):1–12, 2020. → pages 2, 3, 21

Asmussen, S. and Glynn, P. W. *Stochastic simulation: algorithms and analysis*, volume 57. Springer Science & Business Media, 2007. → page 4

Azimzadeh, P. and Forsyth, P. A. Weakly chained matrices, policy iteration, and impulse control. *SIAM Journal on Numerical Analysis*, 54(3):1341–1364, 2016. → page 11

Backenköhler, M., Bortolussi, L., and Wolf, V. Bounding mean first passage times in population continuous-time Markov chains. *arXiv preprint arXiv:1910.12562*, 2019. → page 74

Barton, R. R. and Ivey Jr, J. S. Nelder-Mead simplex modifications for simulation optimization. *Management Science*, 42(7):954–973, 1996. → pages 45, 50

Bellaousov, S., Reuter, J. S., Seetin, M. G., and Mathews, D. H. RNAstructure: web servers for RNA secondary structure prediction and analysis. *Nucleic Acids Research*, 41(W1):W471–W474, 2013. → page 3

Bolhuis, P. G., Chandler, D., Dellago, C., and Geissler, P. L. Transition path sampling: Throwing ropes over rough mountain passes, in the dark. *Annual Review of Physical Chemistry*, 53(1):291–318, 2002. → pages 4, 22, 23

Bonnet, G. *Dynamics of DNA breathing and folding for molecular recognition and computation*. 2000. → pages xiii, xiv, 15, 27, 29, 30, 93, 122

Bonnet, G., Krichevsky, O., and Libchaber, A. Kinetics of conformational fluctuations in DNA hairpin-loops. *Proceedings of the National Academy of Sciences*, 95(15):8602–8606, 1998. → pages xiii, 15, 27, 29, 30, 45, 52, 55, 64, 75, 76, 86, 93, 94, 120, 121

Brand, M. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications*, 415(1):20–30, 2006. → pages 72, 91

Bunch, J. R. and Nielsen, C. P. Updating the singular value decomposition. *Numerische Mathematik*, 31(2):111–129, 1978. → pages 72, 91

Cabriolu, R., Skjelbred Refsnes, K. M., Bolhuis, P. G., and van Erp, T. S. Foundations and latest advances in replica exchange transition interface sampling. *The Journal of Chemical Physics*, 147(15):152722, 2017. → page 22

Cao, Y., Gillespie, D. T., and Petzold, L. R. Adaptive explicit-implicit tau-leaping method with automatic tau selection. *The Journal of Chemical Physics*, 126 (22):224101, 2007. → page 22

Cappelletti, D., Ortiz-Muñoz, A., Anderson, D. F., and Winfree, E. Stochastic chemical reaction networks for robustly approximating arbitrary probability distributions. *Theoretical Computer Science*, 801:64–95, 2020. → pages 3, 7, 95

Chen, S.-J. RNA folding: conformational statistics, folding kinetics, and ion electrostatics. *Annu. Rev. Biophys.*, 37:197–214, 2008. → page 26

Chen, Y.-J., Groves, B., Muscat, R. A., and Seelig, G. Dna nanotechnology from the test tube to the cell. *Nature nanotechnology*, 10(9):748–760, 2015. → pages 1, 15

Cherry, K. M. and Qian, L. Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. *Nature*, 559(7714):370, 2018. → pages 1, 2, 17

Cisse, I. I., Kim, H., and Ha, T. A rule of seven in watson-crick base-pairing of mismatched sequences. *Nature Structural & Moleuclar Biology*, 19(6):623, 2012. → pages 15, 17, 45, 52, 55, 64, 75, 76, 86, 94

Cohen, M. B., Kelner, J., Kyng, R., Peebles, J., Peng, R., Rao, A. B., and Sidford, A. Solving directed laplacian systems in nearly-linear time through sparse LU factorizations. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 898–909. IEEE, 2018. → pages 72, 91

Cohen-Tannoudji, C., Davies, P. C., Diu, B., Laloe, F., Dui, B., et al. *Quantum mechanics*, volume 1. John Wiley & Sons, 1977. → page 74

Dabby, N. L. *Synthetic molecular machines for active self-assembly: prototype algorithms, designs, and experimental study*. PhD thesis, California Institute of Technology, 2013. → pages xiv, 16, 17, 27, 29, 30, 93, 127

Daigle, B. J., Roh, M. K., Petzold, L. R., and Niemi, J. Accelerated maximum likelihood parameter estimation for stochastic biochemical systems. *BMC Bioinformatics*, 13(1):68, 2012. → page 23

Dinh, K. N. and Sidje, R. B. Understanding the finite state projection and related methods for solving the chemical master equation. *Physical Biology*, 13(3): 035003, 2016. → page 24

Dinh, K. N. and Sidje, R. B. An application of the Krylov-FSP-SSA method to parameter fitting with maximum likelihood. *Physical Biology*, 14(6):065001, 2017. → page 24

Donovan, R. M., Sedgewick, A. J., Faeder, J. R., and Zuckerman, D. M. Efficient stochastic simulation of chemical kinetics networks using a weighted ensemble of trajectories. *The Journal of Chemical Physics*, 139(11):09B642_1, 2013. → page 22

Doob, J. L. Topics in the theory of Markoff chains. *Transactions of the American Mathematical Society*, 52(1):37–64, 1942. → pages 4, 9, 12, 27

Doucet, A. and Johansen, A. M. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12(656-704):3, 2009. → pages 22, 51, 67

Duose, D. Y., Schweller, R. M., Zimak, J., Rogers, A. R., Hittelman, W. N., and Diehl, M. R. Configuring robust DNA strand displacement reactions for in situ molecular analyses. *Nucleic Acids Research*, 40(7):3289–3298, 2012. → page 17

Dykeman, E. C. An implementation of the Gillespie algorithm for RNA kinetics with logarithmic time update. *Nucleic Acids Research*, 43(12):5708–5715, 2015. → page 3

Eidelson, N. and Peters, B. Transition path sampling for discrete master equations with absorbing states. *The Journal of Chemical Physics*, 137(9):094106, 2012. → page 23

Elber, R. A new paradigm for atomically detailed simulations of kinetics in biophysical systems. *Quarterly Reviews of Biophysics*, 50, 2017. → page 22

Escobedo, F. A., Borrero, E. E., and Araque, J. C. Transition path sampling and forward flux sampling. Applications to biological systems. *Journal of Physics: Condensed Matter*, 21(33):333101, 2009. → pages 4, 23

Feller, W. *An Introduction to Probability Theory and its Applications*, volume 1. Wiley, New York, 3rd edition, 1968. → pages 68, 69

Flamm, C., Fontana, W., Hofacker, I. L., and Schuster, P. RNA folding at elementary step resolution. *RNA*, 6(03):325–338, 2000. → pages 2, 3, 10, 26

Fletcher, R. Conjugate gradient methods for indefinite systems. In *Numerical Analysis*, pages 73–89. Springer, 1976. → pages 77, 91

Foreman-Mackey, D., Hogg, D. W., Lang, D., and Goodman, J. emcee: The MCMC hammer. *Publications of the Astronomical Society of the Pacific*, 125 (925):306, 2013. → pages 36, 37

Georgoulas, A., Hillston, J., and Sanguinetti, G. Unbiased Bayesian inference for population Markov jump processes via random truncations. *Statistics and Computing*, 27(4):991–1002, 2017. → pages 24, 92

Gibbs, J. and DiMarzio, E. Statistical mechanics of helix-coil transitions in biological macromolecules. *The Journal of Chemical Physics*, 30:271–282, 1959. → page 27

Gillespie, D. T. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977. → pages 4, 9, 12, 27

Gillespie, D. T. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001. → page 22

Gillespie, D. T. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007. → pages 4, 22

Golightly, A. and Wilkinson, D. J. Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*, 1(6), 2011. → page 24

Green, A. A., Silver, P. A., Collins, J. J., and Yin, P. Toehold switches: de-novo-designed regulators of gene expression. *Cell*, 159(4):925–939, 2014. → pages 1, 16, 95

Green, S. J., Lubrich, D., and Turberfield, A. J. DNA hairpins: fuel for autonomous DNA devices. *Biophysical Journal*, 91(8):2966–2975, 2006. → page 16

Hajiaghayi, M., Kirkpatrick, B., Wang, L., and Bouchard-Côté, A. Efficient continuous-time Markov chain estimation. In *International Conference on Machine Learning*, pages 638–646, 2014. → pages 22, 24, 67

Hanke, A. and Metzler, R. Bubble dynamics in DNA. *Journal of Physics A: Mathematical and General*, 36(36):L473, 2003. → page 16

Hansen, L. P. Large sample properties of generalized method of moments estimators. *Econometrica: Journal of the Econometric Society*, pages 1029–1054, 1982. → pages 6, 44, 48, 94

Hansen, L. P., Heaton, J., and Yaron, A. Finite-sample properties of some alternative GMM estimators. *Journal of Business & Economic Statistics*, 14(3): 262–280, 1996. → page 48

Hata, H., Kitajima, T., and Suyama, A. Influence of thermodynamically unfavorable secondary structures on DNA hybridization kinetics. *Nucleic Acids Research*, 46(2):782–791, 2018. → pages 15, 45, 52, 55, 64, 75, 76, 86, 94

Hofacker, I. L. Vienna RNA secondary structure server. *Nucleic Acids Research*, 31(13):3429–3431, 2003. → pages 3, 4, 21, 26

Hordijk, A., Iglehart, D. L., and Schassberger, R. Discrete time methods for simulating continuous time Markov chains. *Advances in Applied Probability*, 8 (4):772–788, 1976. → page 46

Horváth, A. and Manini, D. Parameter estimation of kinetic rates in stochastic reaction networks by the EM method. In *2008 International Conference on BioMedical Engineering and Informatics*, volume 1, pages 713–717. IEEE, 2008. → page 24

Huber, G. A. and Kim, S. Weighted-ensemble brownian dynamics simulations for protein association reactions. *Biophysical Journal*, 70(1):97–110, 1996. → page 22

Jeffreys, H. An invariant form for the prior probability in estimation problems. In *Proceedings of the Royal Society of London a: mathematical, physical and engineering sciences*, volume 186, pages 453–461. The Royal Society, 1946. → page 35

Jennings, E. and Madigan, M. astroABC: an approximate Bayesian computation sequential Monte Carlo sampler for cosmological parameter estimation. *Astronomy and Computing*, 19:16–22, 2017. → page 60

Kavraki, L. E., Svestka, P., Latombe, J.-C., and Overmars, M. H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4):566–580, 1996. → pages 25, 71

Kawasaki, K. Diffusion constants near the critical point for time-dependent Ising models. i. *Physical Review*, 145(1):224, 1966. → page 3

Khodakov, D., Wang, C., and Zhang, D. Y. Diagnostics based on nucleic acid sequence variant profiling: Pcr, hybridization, and ngs approaches. *Advanced Drug Delivery Reviews*, 105:3–19, 2016. → page 16

Kim, J., Doose, S., Neuweiler, H., and Sauer, M. The initial step of DNA hairpin folding: a kinetic analysis using fluorescence correlation spectroscopy. *Nucleic Acids Research*, 34(9):2516–2527, 2006. → pages xiv, 15, 27, 29, 30, 93, 123, 128

Kuehlmann, A., McMillan, K. L., and Brayton, R. K. Probabilistic state space search. In *1999 IEEE/ACM International Conference on Computer-Aided Design. Digest of Technical Papers (Cat. No. 99CH37051)*, pages 574–579. IEEE, 1999. → page 67

Kuntz, J., Thomas, P., Stan, G.-B., and Barahona, M. The exit time finite state projection scheme: bounding exit distributions and occupation measures of continuous-time Markov chains. *SIAM Journal on Scientific Computing*, 41(2): A748–A769, 2019. → pages 5, 74

Kuwahara, H. and Mura, I. An efficient and exact stochastic simulation method to analyze rare events in biochemical systems. *The Journal of Chemical Physics*, 129(16):10B619, 2008. → page 22

Lehmann, E. L. and Casella, G. *Theory of Point Estimation*. Springer Science & Business Media, 2006. → pages 44, 46

Li, S., Jiang, Q., Liu, S., Zhang, Y., Tian, Y., Song, C., Wang, J., Zou, Y., Anderson, G. J., Han, J.-Y., et al. A DNA nanorobot functions as a cancer therapeutic in response to a molecular trigger in vivo. *Nature Biotechnology*, 36 (3):258, 2018. → page 1

Liò, P. and Goldman, N. Models of molecular evolution and phylogeny. *Genome Research*, 8(12):1233–1244, 1998. → pages 3, 8, 95

Lockhart, D. J., Dong, H., Byrne, M. C., Follettie, M. T., Gallo, M. V., Chee, M. S., Mittmann, M., Wang, C., Kobayashi, M., Norton, H., et al. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14(13):1675–1680, 1996. → page 16

Loskot, P., Atitey, K., and Mihaylova, L. Comprehensive review of models and methods for inferences in bio-chemical reaction networks. *arXiv preprint arXiv:1902.05828*, 2019. → page 24

Lück, A. and Wolf, V. Generalized method of moments for estimating parameters of stochastic reaction networks. *BMC Systems Biology*, 10(1):98, 2016. → pages 24, 44

Machinek, R. R., Ouldridge, T. E., Haley, N. E., Bath, J., and Turberfield, A. J. Programmable energy landscapes for kinetic control of DNA strand displacement. *Nature Communications*, 5, 2014. → pages xii, xiii, xiv, 16, 17, 27, 29, 30, 35, 38, 39, 64, 75, 76, 78, 80, 81, 86, 93, 94, 129

Madras, N. N. *Lectures on Monte Carlo methods*, volume 16. American Mathematical Soc., 2002. → pages 22, 67

Marin, J.-M., Pudlo, P., Robert, C. P., and Ryder, R. J. Approximate bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012. → page 60

Mathews, D. H., Sabina, J., Zuker, M., and Turner, D. H. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *Journal of Molecular Biology*, 288(5):911–940, 1999. → pages 4, 20, 26

McGibbon, R. T. and Pande, V. S. Efficient maximum likelihood parameterization of continuous-time Markov processes. *The Journal of Chemical Physics*, 143 (3):034109, 2015. → pages 3, 8, 95

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. → pages 3, 18, 19, 26

Metzner, P., Schütte, C., and Vanden-Eijnden, E. Transition path theory for Markov jump processes. *Multiscale Modeling & Simulation*, 7(3):1192–1219, 2009. → page 74

Morrison, L. E. and Stols, L. M. Sensitive fluorescence-based thermodynamic and kinetic measurements of DNA hybridization in solution. *Biochemistry*, 32(12): 3095–3104, 1993. → pages xiv, 15, 16, 20, 27, 29, 30, 60, 93, 124

Munsky, B. and Khammash, M. The finite state projection algorithm for the solution of the chemical master equation. *The Journal of Chemical Physics*, 124(4):044104, 2006. → pages 5, 24, 73

Muscat, R. A., Bath, J., and Turberfield, A. J. A programmable molecular robot. *Nano Letters*, 11(3):982–987, 2011. → page 16

Nelder, J. A. and Mead, R. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965. → pages 36, 44, 49, 88

Ouldridge, T. E., Louis, A. A., and Doye, J. P. Structural, mechanical, and thermodynamic properties of a coarse-grained DNA model. *The Journal of Chemical Physics*, 134(8):085101, 2011. → pages 2, 3

Pardee, K., Green, A. A., Takahashi, M. K., Braff, D., Lambert, G., Lee, J. W., Ferrante, T., Ma, D., Donghia, N., Fan, M., et al. Rapid, low-cost detection of zika virus using programmable biomolecular components. *Cell*, 165(5): 1255–1266, 2016. → page 1

Parks, M. L., De Sturler, E., Mackey, G., Johnson, D. D., and Maiti, S. Recycling Krylov subspaces for sequences of linear systems. *SIAM Journal on Scientific Computing*, 28(5):1651–1674, 2006. → pages 72, 91

Qian, L. and Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034):1196–1201, 2011. → pages 1, 2, 17

Qian, L., Winfree, E., and Bruck, J. Neural network computation with DNA strand displacement cascades. *Nature*, 475(7356):368, 2011. → page 2

Reimann, P., Schmid, G., and Hänggi, P. Universal equivalence of mean first-passage time and Kramers rate. *Physical Review E*, 60(1):R1, 1999. → page 4

Reynaldo, L. P., Vologodskii, A. V., Neri, B. P., and Lyamichev, V. I. The kinetics of oligonucleotide replacements. *Journal of Moleuclar Biology*, 297(2): 511–520, 2000. → pages xiv, 15, 16, 27, 29, 30, 93, 125

Ripley, B. D. *Stochastic simulation*, volume 316. John Wiley & Sons, 2009. → page 4

Rubino, G. and Tuffin, B. *Rare event simulation using Monte Carlo methods*. John Wiley & Sons, 2009. → pages 4, 22, 67

Sarich, M., Banisch, R., Hartmann, C., and Schütte, C. Markov state models for rare events in molecular dynamics. *Entropy*, 16(1):258–286, 2014. → page 23

Schaeffer, J. M. *Stochastic simulation of the kinetics of multiple interacting nucleic acid strands*. PhD thesis, California Institute of Technology, 2012. → pages 3, 7, 19, 45

Schaeffer, J. M. *Stochastic simulation of the kinetics of multiple interacting nucleic acid strands*. PhD thesis, California Institute of Technology, 2013. → pages 4, 6, 9, 13, 14, 18, 19, 27, 62, 72, 73, 75, 77, 80

Schaeffer, J. M., Thachuk, C., and Winfree, E. Stochastic simulation of the kinetics of multiple interacting nucleic acid strands. In *DNA Computing and Molecular Programming, Lecture Notes in Computer Science*, volume 9211, pages 194–211, 2015. → pages 2, 3, 6, 7, 9, 10, 13, 14, 18, 19, 26, 27, 45, 64, 77

Schnoerr, D., Sanguinetti, G., and Grima, R. Approximation and inference methods for stochastic biochemical kinetics—a tutorial review. *Journal of Physics A: Mathematical and Theoretical*, 50(9):093001, 2017. → page 24

Schreck, J. S., Ouldridge, T. E., Romano, F., Šulc, P., Shaw, L. P., Louis, A. A., and Doye, J. P. DNA hairpins destabilize duplexes primarily by promoting melting rather than by inhibiting hybridization. *Nucleic Acids Research*, 43 (13):6181–6190, 2015. → pages 3, 26

Seeman, N. C. and Sleiman, H. F. Dna nanotechnology. *Nature Reviews Materials*, 3(1):1–23, 2017. → pages 1, 15

Shahabuddin, P. Importance sampling for the simulation of highly reliable Markovian systems. *Management Science*, 40(3):333–352, 1994. → pages 4, 22, 23

Sidje, R. B. and Vo, H. D. Solving the chemical master equation by a fast adaptive finite state projection based on the stochastic simulation algorithm. *Mathematical Biosciences*, 269:10–16, 2015. → page 24

Simmel, F. C., Yurke, B. A. R., and Singh, H. R. Principles and applications of nucleic acid strand displacement reactions. *Chemical Reviews*, 2019. → pages 1, 2, 15

Simmons, G. F. *Differential equations with applications and historical notes*. CRC Press, 1972. → page 74

Singhal, N., Snow, C. D., and Pande, V. S. Using path sampling to build better Markovian state models: predicting the folding rate and mechanism of a tryptophan zipper beta hairpin. *The Journal of Chemical Physics*, 121(1): 415–425, 2004. → pages 4, 23

Soloveichik, D., Cook, M., Winfree, E., and Bruck, J. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633, 2008. → pages 3, 7, 95

Srinivas, N., Ouldridge, T. E., Šulc, P., Schaeffer, J. M., Yurke, B., Louis, A. A., Doye, J. P., and Winfree, E. On the biophysics and kinetics of toehold-mediated DNA strand displacement. *Nucleic Acids Research*, 41(22):10641–10658, 2013. → pages 21, 37, 42, 55, 56

Srinivas, N., Parkin, J., Seelig, G., Winfree, E., and Soloveichik, D. Enzyme-free nucleic acid dynamical systems. *bioRxiv*, page 138420, 2017. → pages 2, 17, 95

Suhov, Y. and Kelbert, M. *Probability and Statistics by Example: Volume 2, Markov Chains: A Primer in Random Processes and Their Applications*, volume 2. Cambridge University Press, 2008. → pages 4, 11

Šulc, P., Romano, F., Ouldridge, T. E., Rovigatti, L., Doye, J. P., and Louis, A. A. Sequence-dependent thermodynamics of a coarse-grained DNA model. *The Journal of Chemical Physics*, 137(13):135101, 2012. → pages 2, 3

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT Press, 2018. → page 67

Tang, X. *Techniques for modeling and analyzing RNA and protein folding energy landscapes*. PhD thesis, Texas A & M University, 2010. → pages 10, 25

Tang, X., Kirkpatrick, B., Thomas, S., Song, G., and Amato, N. M. Using motion planning to study RNA folding kinetics. *Journal of Computational Biology*, 12 (6):862–881, 2005. → pages 2, 3, 25, 71

Thomas, S., Tapia, L., Ekenna, C., Yeh, H.-Y. C., and Amato, N. M. Rigidity analysis for protein motion and folding core identification. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013. → page 71

Thubagere, A. J., Li, W., Johnson, R. F., Chen, Z., Doroudi, S., Lee, Y. L., Izatt, G., Wittman, S., Srinivas, N., Woods, D., et al. A cargo-sorting DNA robot. *Science*, 357(6356), 2017. → pages 2, 17

Tyagi, S. and Kramer, F. R. Molecular beacons: probes that fluoresce upon hybridization. *Nature biotechnology*, 14(3):303–308, 1996. → page 16

Van Kampen, N. G. *Stochastic processes in physics and chemistry*, volume 1. Elsevier, 1992. → page 24

Venkataraman, S., Dirks, R. M., Rothemund, P. W., Winfree, E., and Pierce, N. A. An autonomous polymerization motor powered by DNA hybridization. *Nature Nanotechnology*, 2(8):490–494, 2007. → pages 16, 17

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, pages 1–12, 2020. → pages 72, 77, 88, 91

Wackerly, D., Mendenhall, W., and Scheaffer, R. L. *Mathematical Statistics with Applications*. Cengage Learning, 2014. → page 47

Weinan, E., Ren, W., and Vanden-Eijnden, E. String method for the study of rare events. *Physical Review B*, 66:052301, 2002. → page 22

Weinan, E., Ren, W., and Vanden-Eijnden, E. Finite temperature string method for the study of rare events. *Journal of Physical Chemistry B*, 109(14): 6688–6693, 2005. → pages 22, 67

Wetmur, J. G. Hybridization and renaturation kinetics of nucleic acids. *Annual Review of Biophysics and Bioengineering*, 5(1):337–361, 1976. → pages 16, 45, 52, 55, 94

Wetmur, J. G. and Davidson, N. Kinetics of renaturation of DNA. *Journal of Molecular Biology*, 31(3):349–370, 1968. → page 20

Whitt, W. Continuous-time Markov chains. *Dept. of Industrial Engineering and Operations Research, Columbia University, New York*, 2006. → pages 10, 12

Xayaphoummine, A., Bucher, T., and Isambert, H. Kinefold web server for RNA/DNA folding path and structure prediction including pseudoknots and knots. *Nucleic Acids Research*, 33(suppl 2):W605–W610, 2005. → page 26

Xu, Z. Z. and Mathews, D. H. Experiment-assisted secondary structure prediction with RNAstructure. *RNA Structure Determination: Methods and Protocols*, pages 163–176, 2016. → pages 4, 21, 26

Zadeh, J. N., Steenberg, C. D., Bois, J. S., Wolfe, B. R., Pierce, M. B., Khan, A. R., Dirks, R. M., and Pierce, N. A. NUPACK: analysis and design of nucleic acid systems. *Journal of Computational Chemistry*, 32(1):170–173, 2011. → pages 3, 4, 21, 22, 26, 42

Zhang, D. Y. and Seelig, G. Dynamic dna nanotechnology using strand-displacement reactions. *Nature chemistry*, 3(2):103–113, 2011. → page 2

Zhang, D. Y. and Winfree, E. Control of DNA strand displacement kinetics using toehold exchange. *Journal of the American Chemical Society*, 131(47): 17303–17314, 2009. → pages xiv, 16, 17, 21, 27, 29, 30, 93, 126

Zhang, J. X., Fang, J. Z., Duan, W., Wu, L. R., Zhang, A. W., Dalchau, N., Yordanov, B., Petersen, R., Phillips, A., and Zhang, D. Y. Predicting DNA hybridization kinetics from sequence. *Nature Chemistry*, 10(1):91, 2018. → pages 2, 3, 16, 21, 49, 64, 69, 75, 76, 82, 86, 94

Zolaktaf, S., Dannenberg, F., Rudelis, X., Condon, A., Schaeffer, J. M., Schmidt, M., Thachuk, C., and Winfree, E. Inferring parameters for an elementary step model of DNA structure kinetics with locally context-dependent Arrhenius rates. In *DNA Computing and Molecular Programming, Lecture Notes in Computer Science*, volume 10467, pages 172–187, 2017. → pages vi, 49, 55, 56, 58, 78, 89

Zolaktaf, S., Dannenberg, F., Winfree, E., Bouchard-Côté, A., Schmidt, M., and Condon, A. Efficient parameter estimation for DNA kinetics modeled as continuous-time Markov chains. In *DNA Computing and Molecular Programming, Lecture Notes in Computer Science*, volume 11648, pages 80–99, 2019. → pages vi, 88

Zolaktaf, S., Dannenberg, F., Schmidt, M., Condon, A., and Winfree, E. The pathway elaboration method for mean first passage time estimation in large continuous-time Markov chains with applications to nucleic acid kinetics. In *submission*, 2020. → pages vi, vii

Zuckerman, D. M. and Chong, L. T. Weighted ensemble simulation: review of methodology, applications, and software. *Annual Review of Biophysics*, 46: 43–57, 2017. → page 22

Zuker, M. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Research*, 31(13):3406–3415, 2003. → pages 21, 26

Appendix A

Supplementary for Chapter 3

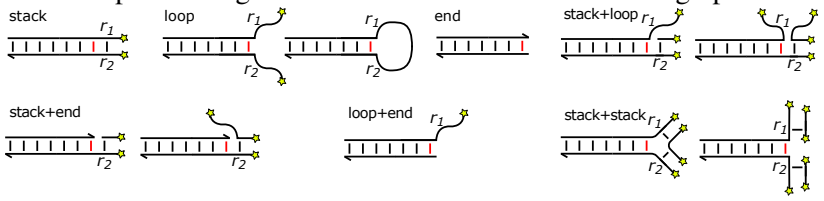
A.1 Local Context

Here we describe how to determine the local context of an elementary step transition, that is, the formation or breakage of a base pair. We use 0-based numbering for numbering bases. That is, in a multi-strand complex, for each strand of length l, the first nucleotide at the 5' end of the strand is numbered 0 and the last nucleotide at the 3' end of the strand is numbered $l-1$.

The local context of a base pair forming or breaking is a pair (l,r), where l and r are the half-contexts on the left and right sides of the base pair forming or breaking, respectively. Each half context is one of seven possibilities: stack, loop, end, stack+loop, stack+end, loop+end, stack+stack. Right half contexts are illustrated in Algorithm 6. Algorithm 6 finds the half contexts, and Algorithm 7 uses Algorithm 6 to find the local context. These algorithms use *dot-parens-plus-mult* notation to represent a secondary structure (state), which is obtained from the *dot-parens-plus* notation. The dot-parens-plus notation uses the symbols '(', ')', '.', '+', and ' '. Matching parentheses represent bases which have formed a base pair, a dot represents a free base pair, and a plus represents a break between strands. For example, '(((((((+)))))))' means that bases 0, 1, 2, 3, 4, and 5 of the first strand are paired with bases 5, 4, 3, 2, 1, and 0 of the second strand, respectively. When all base pairs between strands break, we replace the plus sign by a space. For example, '..... .....' means that no base pair is formed. The dot-parens-plus-mult

Algorithm 6: Find the half context on one side of a base pair forming or breaking

The seven possible right half contexts are illustrated in the graphic below.



Function `HalfContext` (d, f_1, f_2)

Input: d is a dot-parens-plus-mult notation, f_1 and f_2 are either the indices l_1, l_2 that determine the left half context or the indices r_1, r_2 that determine the right half context.

Output: The half context appearing in positions f_1 and f_2.

$c_1 \leftarrow d[f_1]$
$c_2 \leftarrow d[f_2]$
if $c_1 =$ '(' and $c_2 =$ ')' **then**
    counter $\leftarrow 0$
    for k in $[f_1, f_2]$ **do**
        if $d[k] =$ '(' **then** counter $\leftarrow$ counter + 1
        else if $d[k] =$ ')' **then** counter $\leftarrow$ counter - 1
        if counter = 0 **then**
            if $k = f_2$ **then** return stack
            else return stack+stack
else if ($c_1 =$ '(' and $c_2 =$ '(') or ($c_1 =$ ')' and $c_2 =$ ')') or ($c_1 =$ ')' and $c_2 =$ '(') **then** return stack+stack
else if ($c_1 =$ '(' and $c_2 =$ '.') or ($c_1 =$ ')' and $c_2 =$ '.') or ($c_1 =$ '.' and $c_2 =$ ')') or ($c_1 =$ '.' and $c_2 =$ '(') **then** return stack+loop
else if $c_1 =$ '(' and $c_2 =$ '*') or ($c_1 =$ ')' and $c_2 =$ '*') or ($c_1 =$ '*' and $c_2 =$ ')') or ($c_1 =$ '*' and $c_2 =$ '(') **then** return stack+end
else if ($c_1 =$ '.' and $c_2 =$ '*') or ($c_1 =$ '*' and $c_2 =$ '.') **then** return loop+end
else if $c_1 =$ '*' and $c_2 =$ '*' **then** return end
else if $c_1 =$ '.' and $c_2 =$ '.' **then** return loop

notation, inserts '*' at the start and end of a dot-parent-plus notation and before and after all '+' signs and spaces in the dot-parent-plus notation. Thus, '(((((((+)))))))' and '..... .....' change to '*(((((((*+*)))))))*' and '*.....* *.....*', respectively. If two states s_1 and s_2 differ by a single base pair, their dot-parens-plus-mult notation differs at exactly two positions, say e_1 and e_2. The left half context is determined by positions $l_1 = e_1 - 1$ and $l_2 = e_2 + 1$, while the right half context is determined by positions $r_1 = e_1 + 1$ and $r_2 = e_2 - 1$.

Algorithm 7: Find the local context of a base pair forming or breaking

Function `LocalContext` (s_i, s_j)

Input: States s_i and state s_j, which differ by exactly one base pair. (Either of the states can have an extra base pair compared to the other.)

Output: $\langle l, r \rangle$, which is the local context of the base pair breaking or forming in transition from state s_i to state s_j.

$d_i \leftarrow$ dot-parens-plus-mult notation of s_i

$d_j \leftarrow$ dot-parens-plus-mult notation of s_j

$e_1 \leftarrow$ the first position where d_i and d_j differ

$e_2 \leftarrow$ the second position where d_i and d_j differ

$l \leftarrow$ `HalfContext` $(d_i, e_1 - 1, e_2 + 1)$ `// Algorithm 6`

$r \leftarrow$ `HalfContext` $(d_i, e_1 + 1, e_2 - 1)$

return $\langle l, r \rangle$

A.2 Simplified State Spaces

Here we describe `NeighborStates`(s) for a state s in helix association and dissociation, toehold-mediated three-way strand displacement, and toehold-mediated four-way strand exchange. It is used in Algorithm 2 and returns the neighboring states for a state s. In Chapter 3, we describe `NeighborStates`(s) for hairpin closing and opening.

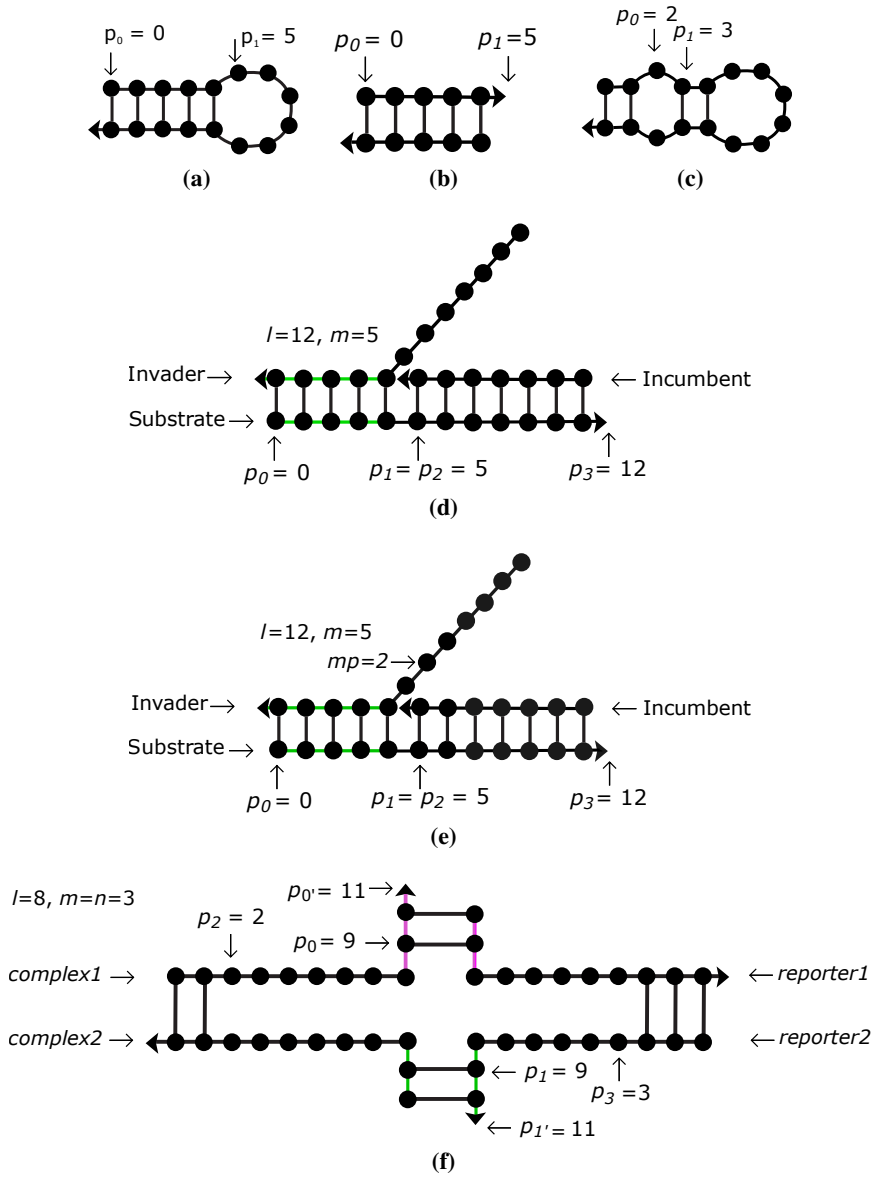


Figure A.1: Examples of simplified state spaces for different type of reactions. (**a**) Hairpin state $s = \langle p_0, p_1 \rangle$. (**b**) Helix state $s = \langle p_0, p_1 \rangle$. (**c**) Bubble state $s = \langle p_0, p_1 \rangle$. (**d**) Toehold-mediated three-way strand displacement state $s = \langle p_0, p_1, p_2, p_3 \rangle$. (**e**) Toehold-mediated three-way strand displacement state $s = \langle p_0, p_1, p_2, p_3 \rangle$ that has a mismatch between the invader and the substrate. (**f**) Toehold-mediated four-way strand exchange state $s = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$.

A.2.1 Helix Association and Dissociation

Let l be the length of a strand in the helix. Each state corresponds to a partial opening of the helix from the left and/or right ends, and is represented by a tuple $\langle p_0, p_1 \rangle$, where $0 \leq p_0 \leq p_1 \leq l$. The tuple indicates that all bases numbered p_0 to $p_1 - 1$ in one strand are paired with bases numbered $l - p_1$ to $l - p_0 - 1$ in the other strand, respectively, and there are no other base pairs in the state. Algorithm 8 generates and returns the set of neighbors of a helix state s. We use the same `NeighborStates`(s) function for helix association and dissociation, however the initial and target states are swapped for these reactions. In helix association, the initial state ($\mathcal{S}_{\text{init}} = \{\langle 0,0 \rangle\}$) has no base pairs. The target state ($\mathcal{S}_{\text{target}} = \{\langle 0,l \rangle\}$) has l base pairs.

Algorithm 8: Generate the neighbor states of a helix state $s = \langle p_0, p_1 \rangle$ (see Figure A.1b)

Function `NeighborStates` ($s = \langle p_0, p_1 \rangle$)
- $\mathcal{N} \leftarrow \emptyset$
- **if** $\langle p_0, p_1 \rangle = \langle 0,0 \rangle$ **then**
  - **for** p in $[0, l-1]$ **do**
    - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1 \rangle$
- **else**
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1 \rangle \cup \langle p_0 + 1, p_1 \rangle \cup \langle p_0, p_1 - 1 \rangle \cup \langle p_0, p_1 + 1 \rangle$
- **foreach** $s' = \langle p'_0, p'_1 \rangle \in \mathcal{N}$ **do**
  - `// The state in which no base pair has formed is shown by` $\langle 0,0 \rangle$
  - **if** $p'_0 = p'_1$ and $p'_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0,0 \rangle$
- **foreach** $s' \in \mathcal{N}$ **do**
  - **if** !`AllowedState`(s') **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$
- return $\mathcal{N}$

Function `AllowedState` ($s' = \langle p_0, p_1 \rangle$)
- **if** $!(0 \leq p_0 \leq p_1 \leq l)$ **then** return False
- return True

A.2.2 Bubble Closing

Let l be the length of the hairpin strand, $m < l/2$ be the length of the stem in the fully closed position, and f be the position where a bubble is formed. Each state corresponds to a partial opening of the bubble from the middle, and is represented by a tuple $\langle p_0, p_1 \rangle$, where $0 < p_0 \leq f \leq p_1 < m$. The tuple indicates that all bases numbered 0 to $p_0 - 1$ are paired with bases numbered $l - p_0$ to $l - 1$, respectively, and all bases numbered p_1 to $m - 1$ are paired with bases numbered $l - m$ to $l - p_1 - 1$, respectively, and there are no other base pairs in the state. Algorithm 9 generates and returns the set of neighbors of a bubble closing state s. In the initial state ($\mathcal{S}_{\text{init}} = \{\langle f, f+1 \rangle\}$), all base pairs in the hairpin stem have formed except for a bubble of size 1 in the stem at position f. In the target state ($\mathcal{S}_{\text{target}} = \{\langle f, f \rangle\}$), all base pairs have formed.

Algorithm 9: Generate the neighbor states of a bubble state $s = \langle p_0, p_1 \rangle$ (see Figure A.1c)

Function `NeighborStates` ($s = \langle p_0, p_1 \rangle$)
- $\mathcal{N} \leftarrow \emptyset$
- $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1 \rangle \cup \langle p_0 + 1, p_1 \rangle \cup \langle p_0, p_1 - 1 \rangle \cup \langle p_0, p_1 + 1 \rangle$
- **foreach** $s' = \langle p'_0, p'_1 \rangle \in \mathcal{N}$ **do**
  - `// The state in which all base pairs have formed is shown by` $\langle f, f \rangle$
  - **if** $p'_0 = p'_1$ and $p'_0 \neq f$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle f, f \rangle$
- **foreach** $s' \in \mathcal{N}$ **do**
  - **if** !`AllowedState` (s') **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$
- return $\mathcal{N}$

Function `AllowedState` ($s' = \langle p_0, p_1 \rangle$)
- **if** $!(0 < p_0 \leq f \leq p_1 < m)$ **then** return False
- return True

A.2.3 Toehold-Mediated Three-Way Strand Displacement

Let l be the length of the substrate. For simplicity, let l also be the length of the invader, m be the toehold length, and $l-m$ be the length of the incumbent. Each state is represented by a tuple $\langle p_0, p_1, p_2, p_3 \rangle$, where $0 \leq p_0 \leq p_1 \leq p_2 \leq p_3 \leq l$ and $p_2 \geq m$. The tuple indicates that all bases numbered p_0 to $p_1 - 1$ in the substrate are paired with bases numbered $l - p_1$ to $l - p_0 - 1$ in the invader, respectively, all bases numbered p_2 to $p_3 - 1$ in the substrate are paired with bases numbered $l - p_3$ to $l - p_2 - 1$ in the incumbent, respectively, and there are no other base pairs in the state. Algorithm 10 generates and returns the set of neighbors of a toehold-mediated three-way strand displacement state s. In the initial state ($\mathcal{S}_{\text{init}} = \{\langle 0, 0, m, l \rangle\}$), the substrate is completely attached to the incumbent, but completely detached from the invader. In the target state ($\mathcal{S}_{\text{target}} = \{\langle 0, l, l, l \rangle\}$), the substrate is completely detached from the incumbent, but completely attached to the invader. Algorithm 11 adapts algorithm 10 for toehold-mediated three-way strand displacement with mismatches between the invader and the substrate. In the algorithm, *mp* is a pointer to the mismatch position in the displacement domain.

Note that in algorithms 10 and 11, to efficiently obtain mean first passage times with sparse matrix computations, we further heuristically prune the state space of each reaction (described in the algorithms).

A.2.4 Toehold-Mediated Four-way Strand Exchange

Let *complex* be the first helix and *complex1* and *complex2* be the two strands in this helix. Let *reporter* be the second helix and *reporter1* be the strand in this helix that is complementary to *complex1* and *reporter2* be the strand in this helix that is complementary to *complex2*. Let l be the length of the helices excluding their toehold. For simplicity, let *m* be the toehold length of *complex1* and *reporter1* and let *n* be the toehold length of *complex2* and *reporter2*. Each state is represented by a tuple $\langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$. The tuple indicates that all bases numbered p_0 to $p_{0'} - 1$ in *complex1* have paired with bases numbered $l + m - p_{0'}$ to $l + m - p_0 - 1$ in *reporter1*, respectively, all bases numbered 0 to $p_2 - 1$ in *complex1* are paired with bases numbered $l + n - p_2$ to $l + n - 1$ in *complex2*, respectively, all bases numbered p_1 to $p_{1'} - 1$ in *reporter2* are paired with bases numbered $l + n -$

Algorithm 10: Generate the neighbor states of a toehold-mediated three-way strand displacement state $s = \langle p_0, p_1, p_2, p_3 \rangle$ (see Figure A.1d)

Function `NeighborStates` ($s = \langle p_0, p_1, p_2, p_3 \rangle$)

- $\mathcal{N} \leftarrow \emptyset$
- **if** $p_0 = p_1$ **then**
  - `// If the invader and the substrate are detached, they can form a base pair`
  - **for** p in $[0, p_2 - 1]$ **do**
    - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1, p_2, p_3 \rangle$
- **else**
  - `// The invader and substrate can form/break a base pair`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1, p_2, p_3 \rangle \cup \langle p_0 + 1, p_1, p_2, p_3 \rangle \cup \langle p_0, p_1 - 1, p_2, p_3 \rangle \cup \langle p_0, p_1 + 1, p_2, p_3 \rangle$
  - `// The incumbent and substrate can form/break a base pair`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_2 - 1, p_3 \rangle \cup \langle p_0, p_1, p_2 + 1, p_3 \rangle \cup \langle p_0, p_1, p_2, p_3 - 1 \rangle \cup \langle p_0, p_1, p_2, p_3 + 1 \rangle$
- **foreach** $s' = \langle p'_0, p'_1, p'_2, p'_3 \rangle \in \mathcal{N}$ **do**
  - `// States in which the substrate and invader are detached are shown by` $\langle 0, 0, p'_2, p'_3 \rangle$
  - **if** $p'_0 = p'_1$ and $p'_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0, p'_2, p'_3 \rangle$
  - `// States in which the substrate and incumbent are detached are shown by` $\langle p'_0, p'_1, l, l \rangle$
  - **if** $p'_2 = p'_3$ and $p'_2 \neq l$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle p'_0, p'_1, l, l \rangle$
- **foreach** $s' \in \mathcal{N}$ **do**
  - **if** !`AllowedState(`s'`)` **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$
- return $\mathcal{N}$

Function `AllowedState` ($s' = \langle p_0, p_1, p_2, p_3 \rangle$)

- **if** $!(0 \leq p_0 \leq p_1 \leq p_2 \leq p_3 \leq l$ and $p_2 \geq m)$ **then** return False
- `// Heuristically, further prune the state space`
- **if** $p_0 = p_1$ and $p_2 = p_3$ **then** return False
- `// Disallow the complex to dissociate into three strands`
- **if** $(p_2 - p_1 > 1$ or $p_0 \neq 0)$ and $(0 < m < p_2)$ **then** return False
- **if** $p_2 - p_1 > m + 2$ **then** return False
- return True

Algorithm 11: Generate the neighbor states of a toehold-mediated three-way strand displacement state $s = \langle p_0, p_1, p_2, p_3 \rangle$ that has a mismatch between the invader and the substrate (see Figure A.1e)

Function `NeighborStates` ($s = \langle p_0, p_1, p_2, p_3 \rangle$)
- $\mathcal{N} \leftarrow \emptyset$
- **if** $p_0 = p_1$ **then**
  - `// If the invader and the substrate are detached, they can form a base pair`
  - **for** p in $[0, p_2 - 1]$ **do**
    - **if** $p \neq m + mp - 1$ **then**
      - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1, p_2, p_3 \rangle$
- **else**
  - `// The incumbent and substrate can form/break a base pair`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_2 - 1, p_3 \rangle \cup \langle p_0, p_1, p_2 + 1, p_3 \rangle \cup \langle p_0, p_1, p_2, p_3 - 1 \rangle \cup \langle p_0, p_1, p_2, p_3 + 1 \rangle$
  - `// The invader and substrate can form/break a base pair`
  - **if** $p_0 - 1 \neq m + mp - 1$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 2, p_1, p_2, p_3 \rangle$
  - **if** $p_0 + 1 \neq m + mp - 1$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 + 1, p_1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 + 2, p_1, p_2, p_3 \rangle$
  - **if** $p_1 - 1 \neq m + mp - 1$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 - 1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 - 2, p_2, p_3 \rangle$
  - **if** $p_1 + 1 \neq m + mp - 1$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 + 1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 + 2, p_2, p_3 \rangle$
- **foreach** $s' = \langle p'_0, p'_1, p'_2, p'_3 \rangle \in \mathcal{N}$ **do**
  - `//` $\langle 0, 0, p'_2, p'_3 \rangle$ `means the substrate and invader are detached`
  - **if** $p'_0 = p'_1$ and $p'_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0, p'_2, p'_3 \rangle$
  - `//` $\langle p'_0, p'_1, l, l \rangle$ `means the substrate and incumbent are detached`
  - **if** $p'_2 = p'_3$ and $p'_2 \neq l$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle p'_0, p'_1, l, l \rangle$
- **foreach** $s' \in \mathcal{N}$ **do**
  - **if** !`AllowedState(`s'`)` **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$
- return $\mathcal{N}$

Function `AllowedState` ($s' = \langle p_0, p_1, p_2, p_3 \rangle$)
- **if** $!(0 \leq p_0 \leq p_1 \leq p_2 \leq p_3 \leq l$ and $p_2 \geq m)$ **then** return False
- `// Heuristically, further prune the state space`
- **if** $p_0 = p_1$ and $p_2 = p_3$ **then** return False
- `// Disallow the complex to dissociate into three strands`
- **if** $(p_2 - p_1 > 5$ or $p_0 \neq 0)$ and $(0 < m < p_2)$ **then** return False
- **if** $p_2 - p_1 > m + 4$ **then** return False
- return True

$p_{1'}$ to $l+n-p_1-1$ in *complex2*, respectively, all bases numbered 0 to p_3-1 in *reporter2* are paired with bases numbered $l+m-p_3$ to $l+m-1$ in *reporter1*, respectively, and there are no other base pairs in the state. Algorithm 12 generates and returns the set of neighbors of a toehold-mediated four-way strand exchange state s. In the initial state ($\mathcal{S}_{\text{init}} = \{\langle l+m, l+n, l+m, l+n, l, l\rangle\}$), *complex1* and *complex2* are completely bound except in their toeholds (have formed the *complex* helix), *reporter1* and *reporter2* are completely bound except in their toeholds (have formed the *reporter* helix), and no base pairs have formed between the *complex* helix and the *reporter* helix. Hence, each helix has two complementary strands except for their toeholds. In the target state ($\mathcal{S}_{\text{target}} = \{\langle 0, 0, l+m, l+n, 0, 0\rangle\}$), the *reporter* and *complex* helices have completely exchanged strands and two new helices, which have complementary strands, are formed.

Note that in algorithm 12, to efficiently obtain mean first passage times with sparse matrix computations, we further heuristically prune the state space of each reaction (described in the algorithm).

Algorithm 12: Generate the neighbor states of a toehold-mediated four-way strand exchange state $s = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$ (see Figure A.1f)

Function `NeighborStates` ($s = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$)

- **if** $p_0 = p_{0'}$ **then** `// If complex1 and reporter1 are detached, they can form a base pair`
  - **for** p in $[\max\{p_2, p_3\}, l+m-1]$ **do**
    - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p_1, p+1, p_{1'}, p_2, p_3 \rangle$
- **else** `// complex1 can form/break a base pair with reporter1`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0 + 1, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'} - 1, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'} + 1, p_{1'}, p_2, p_3 \rangle$
- **if** $p_1 = p_{1'}$ **then** `// If reporter2 and complex2 are detached, they can form a base pair`
  - **for** p in $[\max\{p_2, p_3\}, l+n-1]$ **do**
    - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p, p_1, p+1, p_2, p_3 \rangle$
- **else** `// reporter2 can form/break a base pair with complex2`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 - 1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1 + 1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'} - 1, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'} + 1, p_2, p_3 \rangle$
- **if** $(p_0 \neq p_{0'}$ or $p_1 \neq p_{1'})$ or $(m = 0$ or $n = 0)$ **then**
  - `// If complex1 and reporter1 are attached or complex2 and reporter2 are attached or a toehold does not exist, then complex1 and complex2 can form/break a base pair`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2 - 1, p_3 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2 + 1, p_3 \rangle$
  - `// If complex1 and reporter1 are attached or complex2 and reporter2 are attached or a toehold does not exist, then reporter1 and reporter2 can form/break a base pair`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 - 1 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 + 1 \rangle$
- **foreach** $s' = \langle p'_0, p'_1, p'_{0'}, p'_{1'}, p'_2, p'_3 \rangle \in \mathcal{N}$ **do**
  - `// States in which complex1 and reporter1 are detached are shown by` $\langle l+m, p'_1, l+m, p'_{1'}, p'_2, p'_3 \rangle$
  - **if** $p'_0 = p'_{0'}$ and $0 \leq p'_0 < l+m$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle l+m, p'_1, l+m, p'_{1'}, p'_2, p'_3 \rangle$
  - `// States in which reporter2 and complex2 are detached are shown by` $\langle p'_0, l+n, p'_{0'}, l+n, p'_2, p'_3 \rangle$
  - **if** $p'_1 = p'_{1'}$ and $0 \leq p'_1 < l+n$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle p'_0, l+n, p'_{0'}, l+n, p'_2, p'_3 \rangle$
- **foreach** $s' \in \mathcal{N}$ **do**
  - **if** `!AllowedState(`s'`)` **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$
- return $\mathcal{N}$

Function `AllowedState` ($s' = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$)

- **if** $!(p_3 \leq p_0$ and $p_3 \leq p_1$ and $p_2 \leq p_0$ and $p_2 \leq p_1$ and $0 \leq p_2 \leq l$ and $0 \leq p_3 \leq l$ and $0 \leq p_0 \leq p_{0'} \leq l+m$ and $0 \leq p_1 \leq p_{1'} \leq l+n)$ **then** return False
- `// Heuristically, further prune the state space`
- **if** $(p_0 = p_{0'}$ or $p_1 = p_{1'})$ and $(p_2 = 0$ or $p_3 = 0)$ **then** return False
- `// Disallow the complex to dissociate into three or four complexes`
- **if** $(m = 0$ or $n = 0)$ and $(p_0 = p_{0'}$ or $p_1 = p_{1'})$ and $(l - p_2 > 3 - m/3$ or $l - p_3 > 3 - n/3)$ **then** return False
- **if** $(m \neq 0$ and $n \neq 0)$ and $(p_0 = p_{0'}$ or $p_1 = p_{1'})$ and $(p_2 < l-1$ or $p_3 < l-1)$ **then** return False
- **if** $p_{0'} < l$ or $p_{1'} < l$ **then** return False
- **if** $(p_0 \neq p_{0'}$ and $p_1 \neq p_{1'})$ and $(|p_0 - p_3| + |p_0 - p_2| + |p_1 - p_3| + |p_1 - p_2| > 8 - n/3 - m/3)$ **then** return False
- return True

A.3 Experimental Plot Reproduction

The following plots show the performance of the Metropolis and the Arrhenius models on the training and testing datasets. Dashed lines indicate model fits and predictions and solid lines indicate experimentally determined values. For the MCMC ensemble method, error bars indicate the range (minimum to maximum) of predictions.

A.3.1 Training Set ($\mathcal{D}_{\text{train}}$)

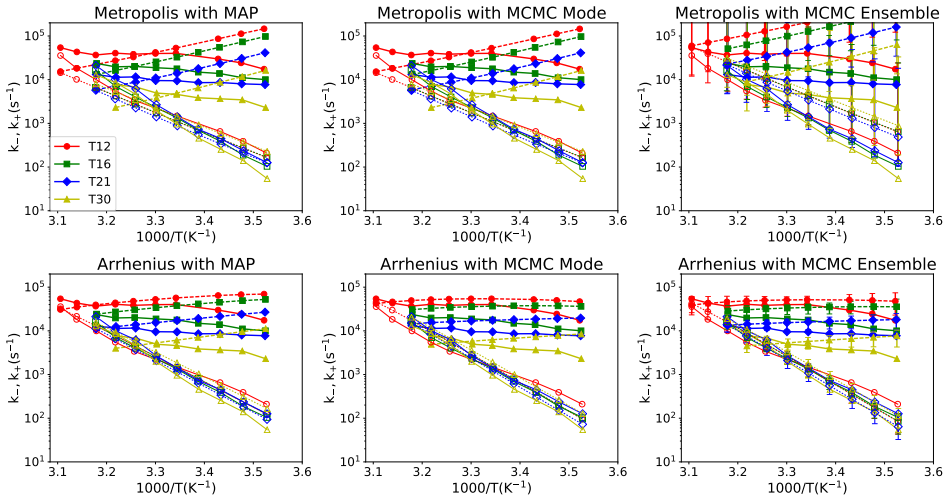


Figure A.2: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open) with sequence $5'$-$CCCAA$-$(T)_n$-$TTGGG$-$3'$ where n is 12,16, 21, or 30, experimental data (solid lines) from Figure 4 of Bonnet et al. (1998).

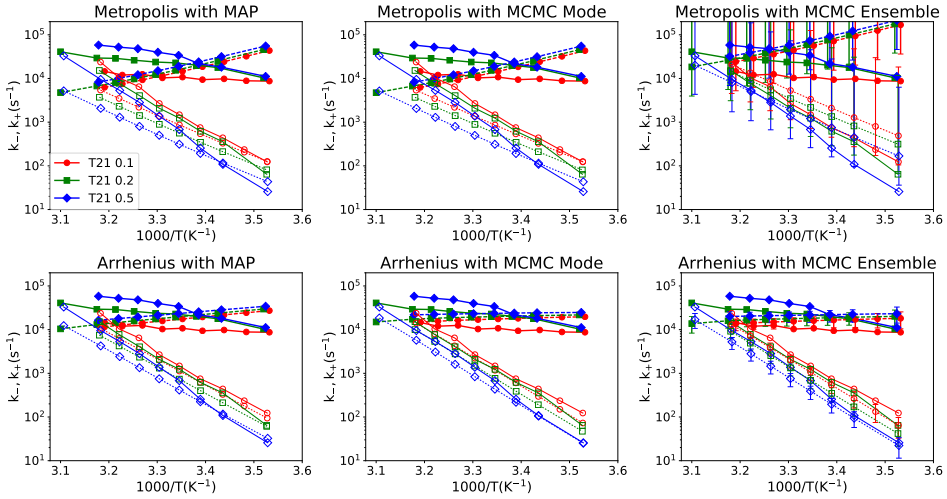


Figure A.3: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open) with sequence $5'$-*CCCAA*-$(T)_{21}$-*TTGGG*-$3'$ at different salt concentrations, experimental data (solid lines) from Figure 6 of Bonnet et al. (1998). Figure 6 of Bonnet et al. (1998) wrongfully notes the use of a poly-A instead of a poly-T hairpin loop, which becomes evident in comparison to Figure 5 of the same work (private communication with the authors).

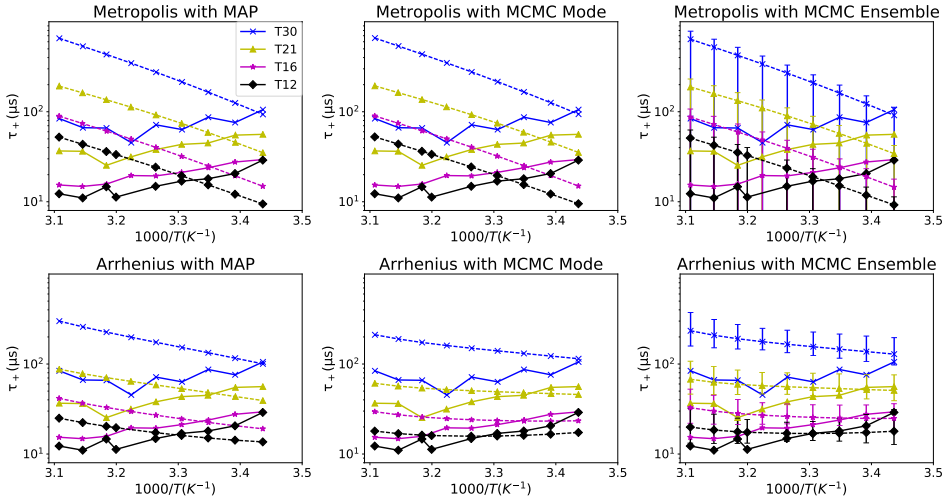


Figure A.4: Model fitting (dashed lines) of reaction timescales (y axis) for hairpin closing with sequence 5′-*CCCAA*-$(T)_n$-*TTGGG*-3′ where n is 12,16, 21, or 30, experimental data (solid lines) from Figure 3.28 of Bonnet (2000).

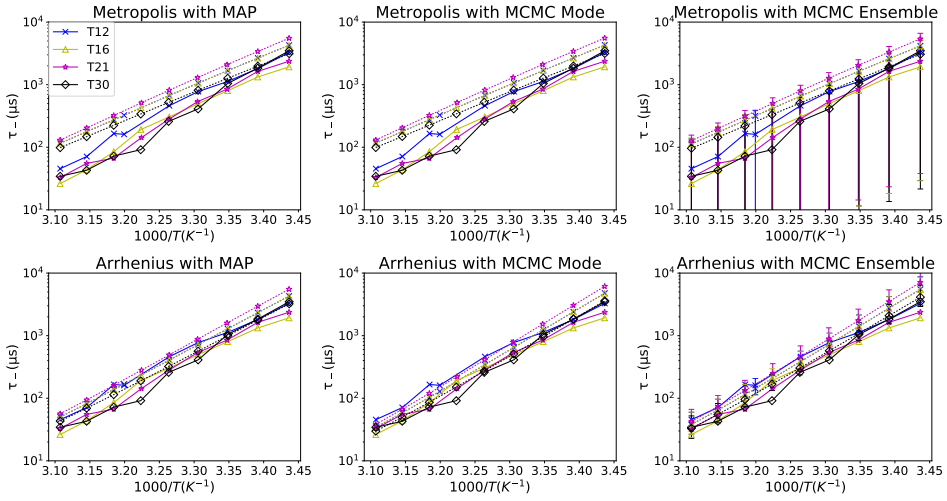


Figure A.5: Model fitting (dashed lines) of reaction timescales (y axis) for hairpin opening with sequence 5′-*CCCAA*-$(T)_n$-*TTGGG*-3′ where n is 12,16, 21, or 30, experimental data (solid lines) from Figure 3.28 of Bonnet (2000).

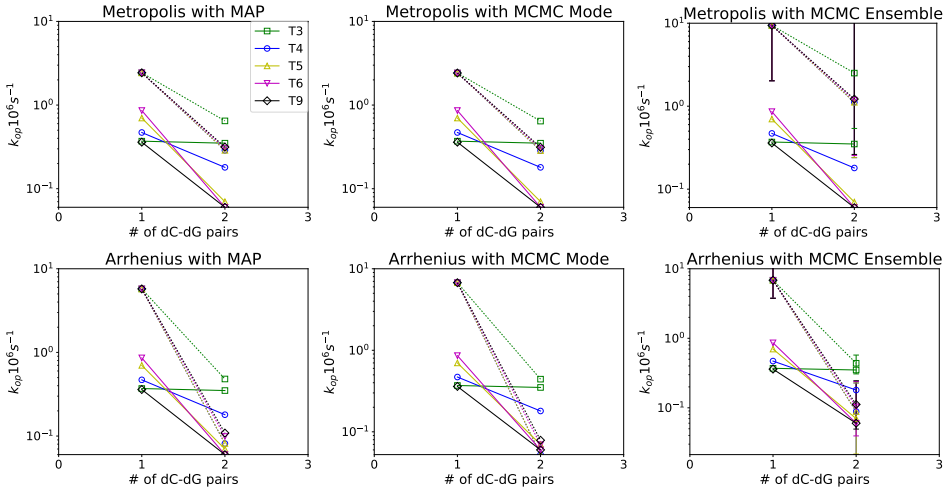


Figure A.6: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin opening with sequence F-$(dC)_y$-$(dT)_x$-$(dG)_y$ (x ranging from 3 to 9) as a function of dC-dG pairs (y ranging from 1 to 2), experimental data (solid lines) from Table 1 (Figure 3b) of Kim et al. (2006).

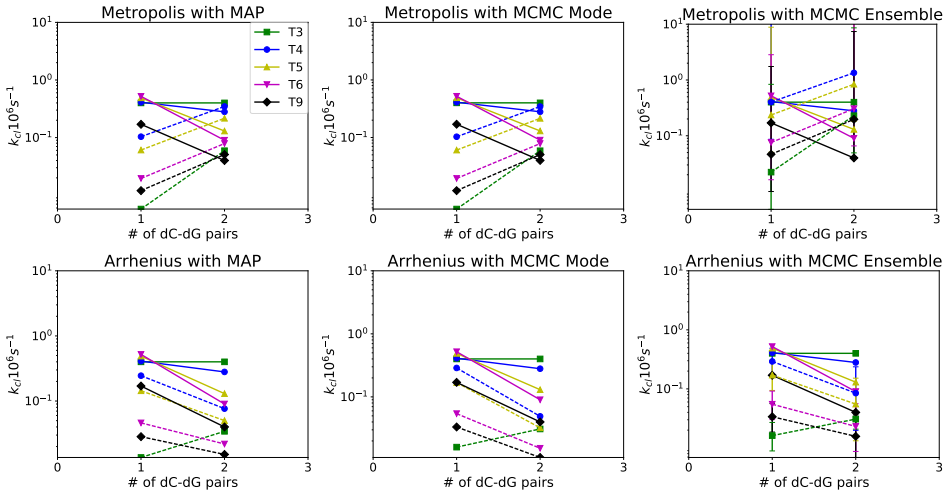


Figure A.7: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin closing with sequence F-$(dC)_y$-$(dT)_x$-$(dG)_y$ (x ranging from 3 to 9) as a function of dC-dG pairs (y ranging from 1 to 2), experimental data (solid lines) from Table 1 (Figure 3b) of Kim et al. (2006).

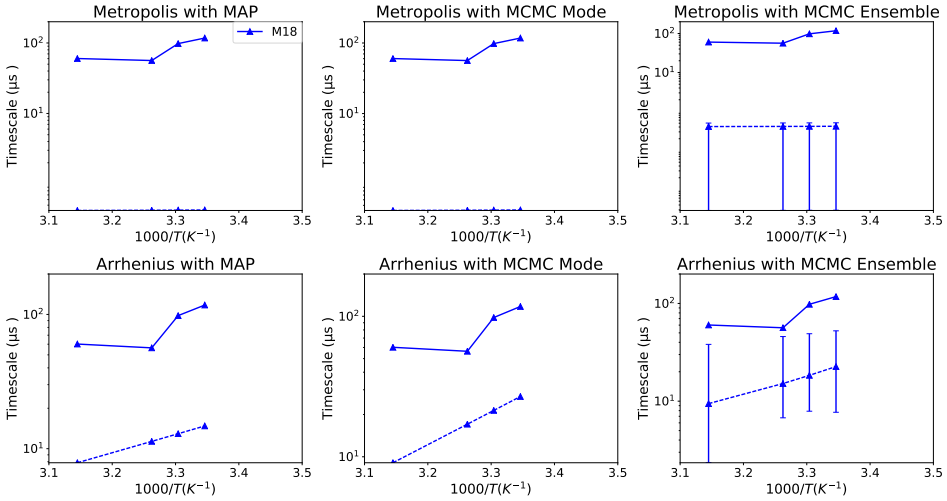


Figure A.8: Model fitting (dashed lines) of reaction timescales (y axis) for bubble closing with sequence M_{18}, experimental data (solid lines) from Figure 4 of Altan-Bonnet et al. (2003).

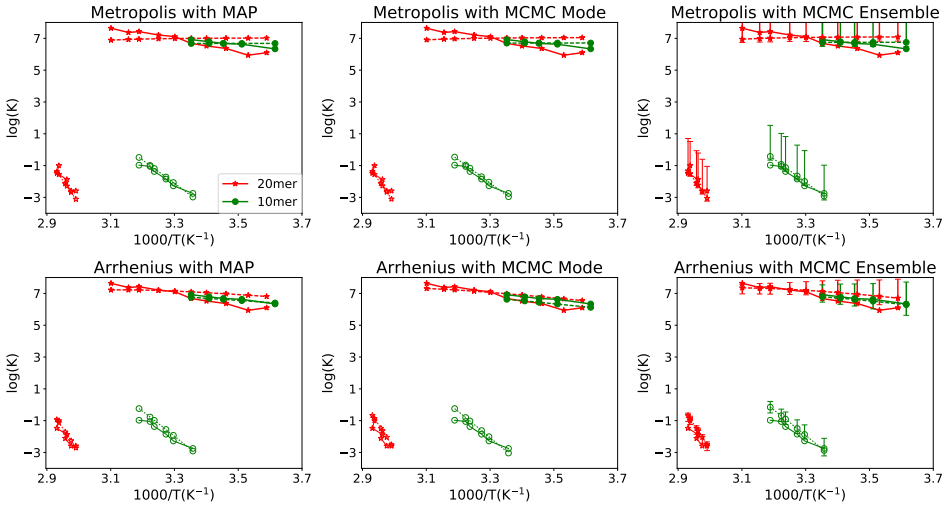


Figure A.9: Model fitting (dashed lines) of reaction rate constants (y axis) for helix association (solid) and disassociation (solid), experimental data (solid lines) from Figure 6 of Morrison and Stols (1993). 10mer and 20mer are variation in the length of the strand.

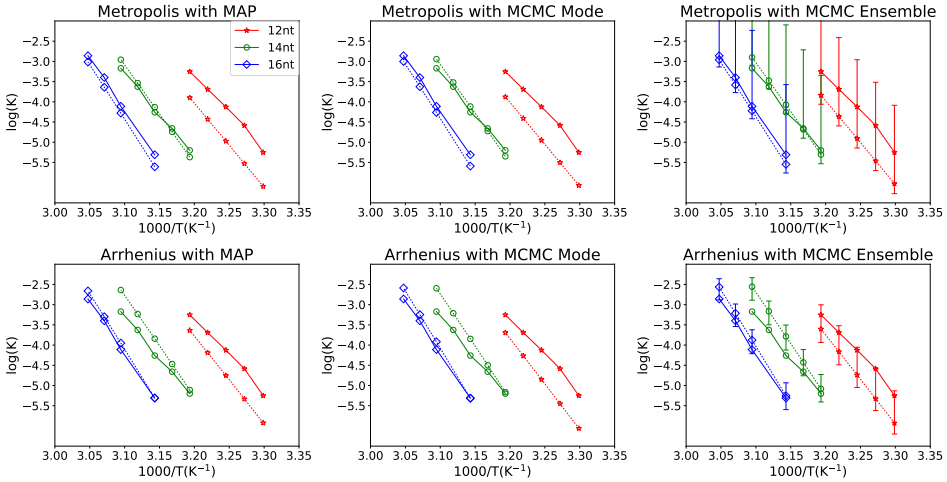


Figure A.10: Model fitting (dashed lines) of reaction rate constants (y axis) for helix disassociation, experimental data (solid lines) from Figure 6 of Reynaldo et al. (2000). 12nt, 14nt, and 16nt are variations in the length of the strand.

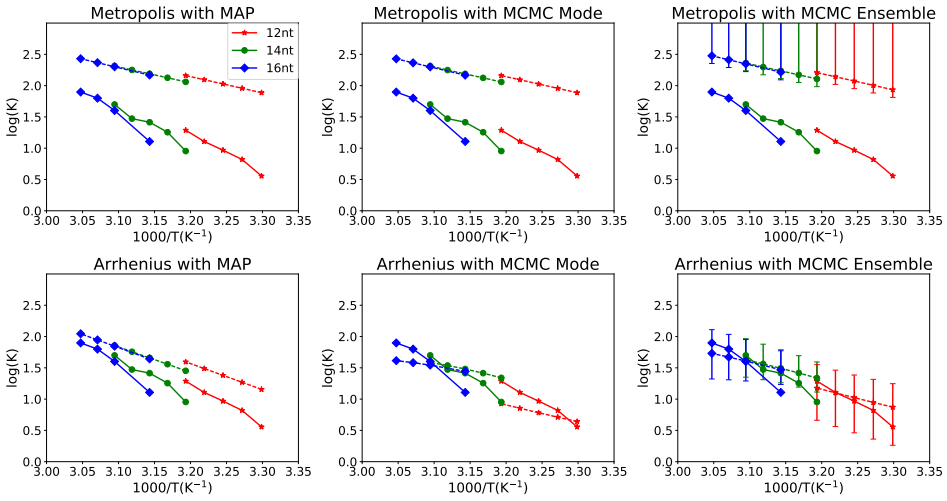


Figure A.11: Model fitting (dashed lines) of reaction rate constants (y axis) for toehold-mediated three-way strand displacement, experimental data (solid lines) from Figure 6 of Reynaldo et al. (2000). 12nt, 14nt, and 16nt are variations in the length of the strand.

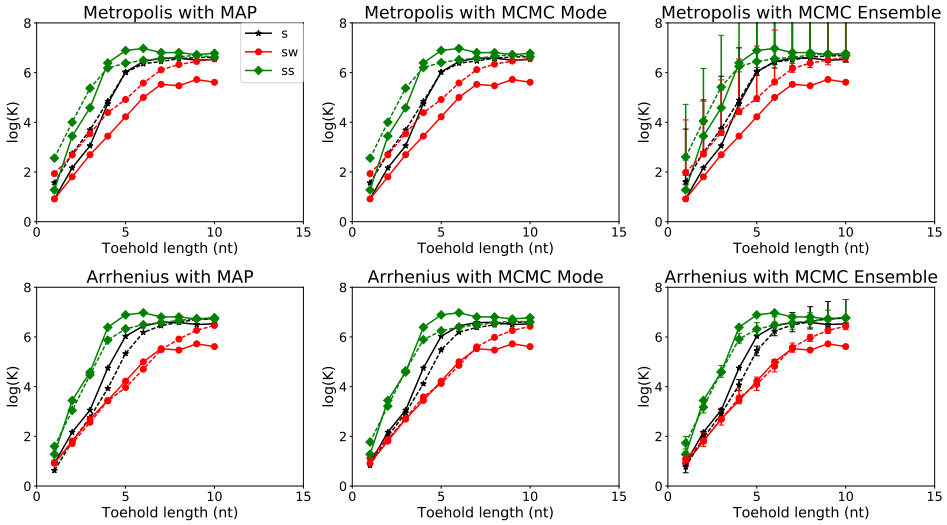


Figure A.12: Model fitting (dashed lines) of reaction rate constants (y axis) for toehold-mediated three-way strand displacement, experimental data (solid lines) from Figure 3b of Zhang and Winfree (2009). The toehold is varied between strong (ss), regular (s) and weak (sw) binding strength by varying the G/C content of the toehold sequence.

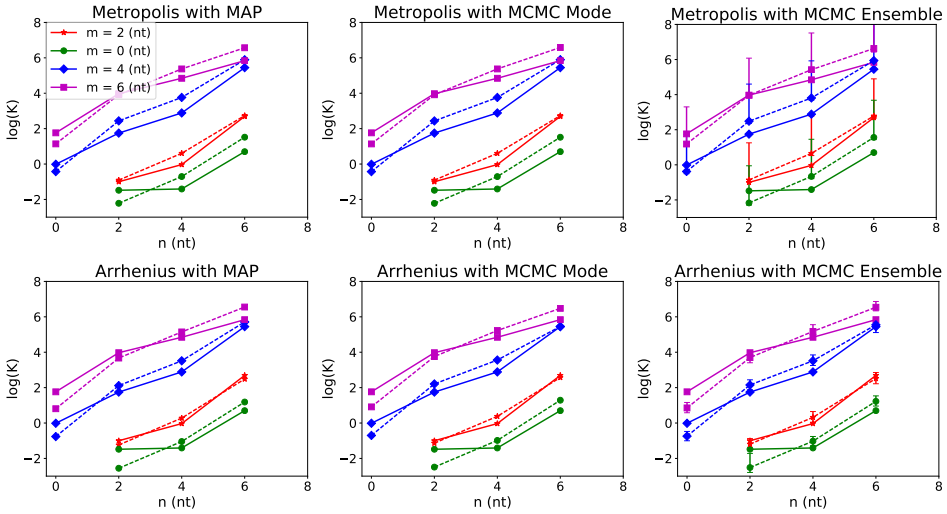


Figure A.13: Model fitting (dashed lines) of reaction rate constants (y axis) for toehold-mediated four-way strand exchange, experimental data (solid lines) from Table 5.2 of Dabby (2013). m (shown on the legend) and n (shown on the x-axis) are variations in the length of the toehold domains (see Appendix A.2.4).

A.3.2 Testing Set ($\mathcal{D}_{\text{test}}$)

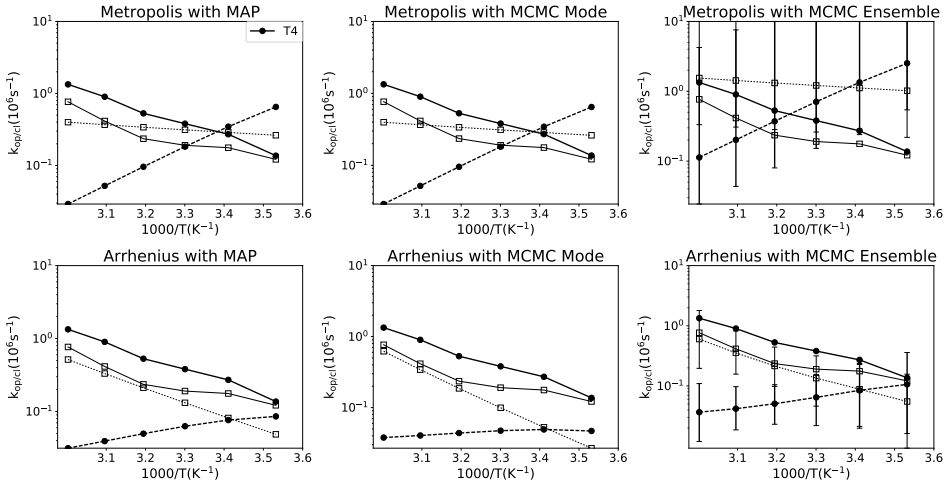


Figure A.14: Model predictions (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open), experimental data (solid lines) from Figure 5a of Kim et al. (2006).

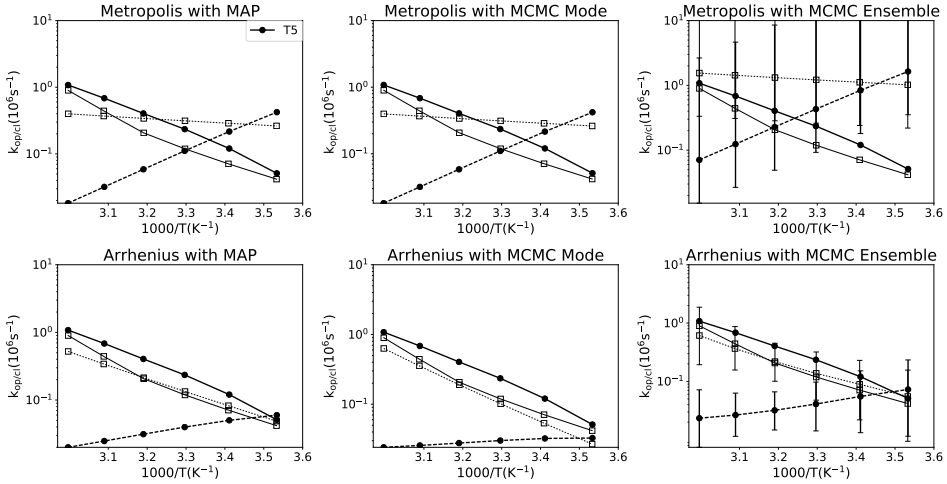


Figure A.15: Model predictions (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open), experimental data (solid lines) from Figure 5b of Kim et al. (2006).

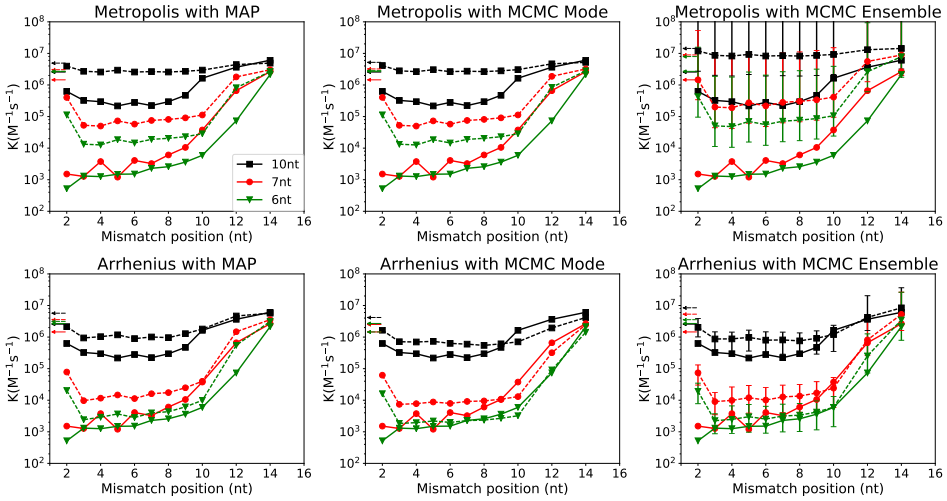


Figure A.16: Model predictions (dashed lines) of reaction rate constants (y axis) for toehold-mediated three-way strand displacement with mismatches, experimental data (solid lines) from Figure 2d of Machinek et al. (2014). Arrows indicate no mismatch. The mismatch in the invading strand affects the reaction rate. The length of the toehold domain is ten, seven, and six nucleotides long for ■, ●, and ▼, respectively.

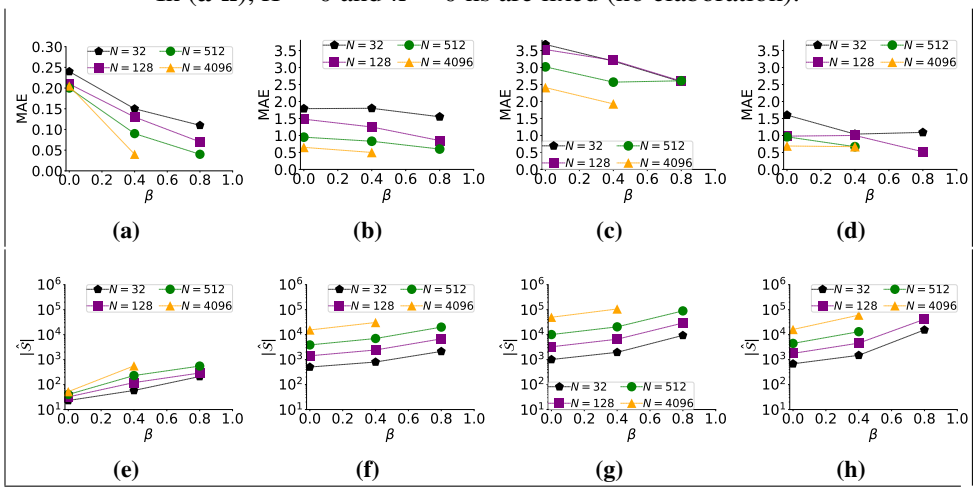
Appendix B

Supplementary for Chapter 5

B.1 The Mean Absolute Error of the Pathway Elaboration Method for Nucleic Acid Kinetics

Figures B1 and B2 represent Figure 5.5 from the main text by varying only two parameters at a time. See the main text for the explanation of these figures.

In (**a**-**h**), $K = 0$ and $\kappa = 0$ ns are fixed (no elaboration).



In (**a**-**h**), $K = 256$ and $\kappa = 16$ ns are fixed.

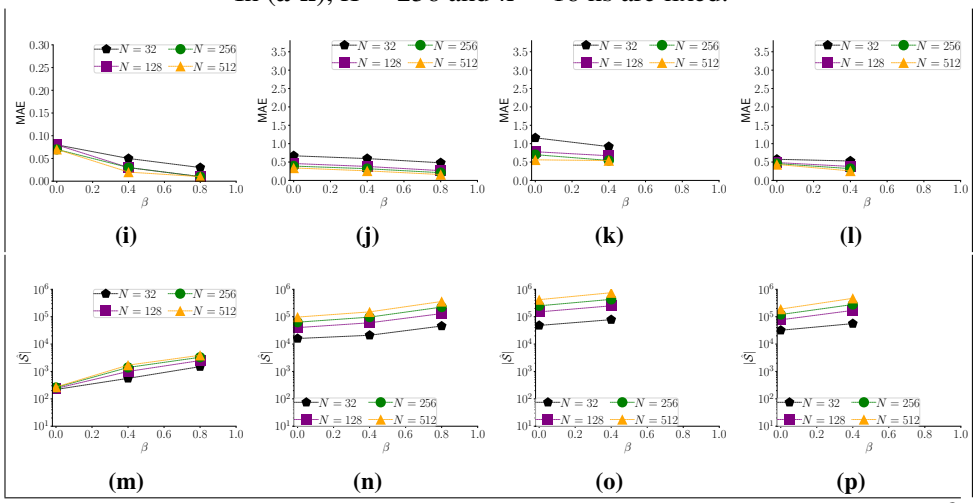
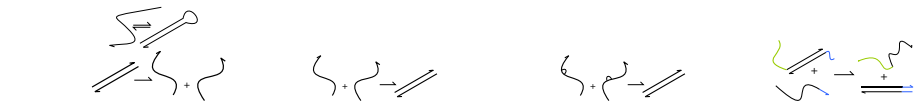
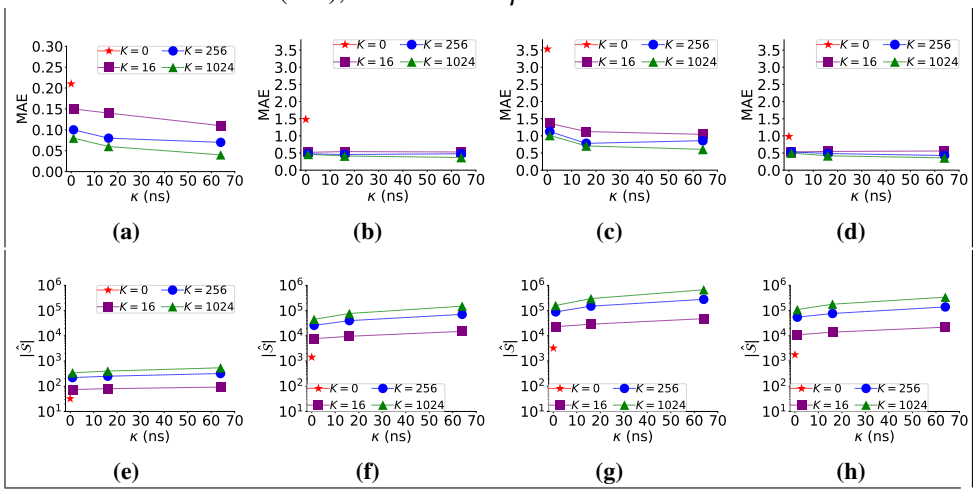


Figure B1: The effect of pathway construction with different values of N and β and fixed values of K and κ. MAE for (**a**) datasets No. 1,2, and 3, (**b**) dataset No. 4, (**c**) dataset No. 5, and (**d**) dataset No. 6. $|\mathcal{S}|$ for (**e**) datasets No. 1,2, and 3, (**f**) dataset No. 4, (**g**) dataset No. 5, and (**h**) dataset No. 6. For the missing settings, pathway elaboration did not finish within two weeks computation time.



In (**a**-**h**), $N = 128$ and $\beta = 0.0$ are fixed.



In (**i**-**p**), $N = 128$ and $\beta = 0.6$ are fixed.

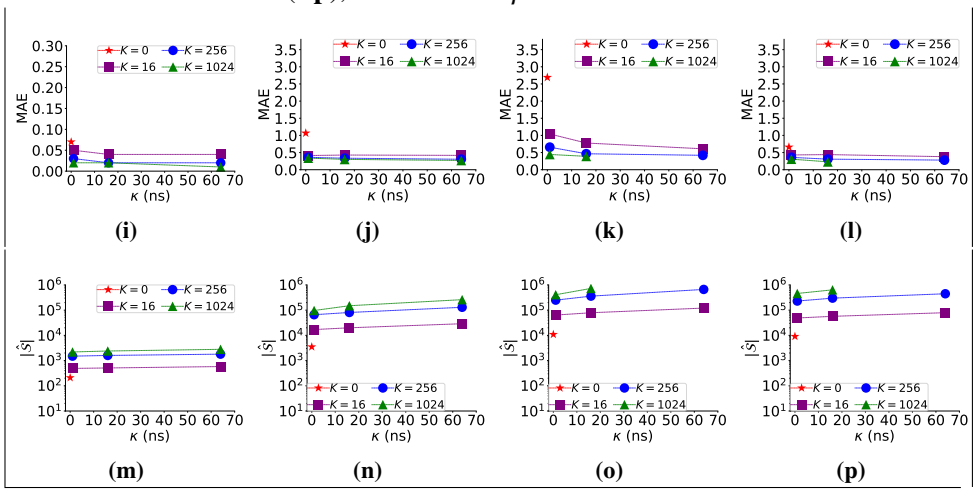


Figure B2: The effect of state elaboration, with different values of K and κ and fixed values of N and β. $N = 0$ indicates that the states of the pathway are not elaborated. (**a**), (**e**), (**i**), and (**m**) correspond to datasets No. 1,2, and 3. (**b**), (**f**), (**j**), and (**n**) correspond to dataset No. 4. (**c**) (**g**), (**k**), and (**o**) correspond to dataset No. 5. (**d**), (**h**), (**l**), and (**p**) correspond to dataset No. 6. For the missing settings, pathway elaboration did not finish within two weeks computation time.