Design and Analysis of Nucleic Acid Reaction Pathways

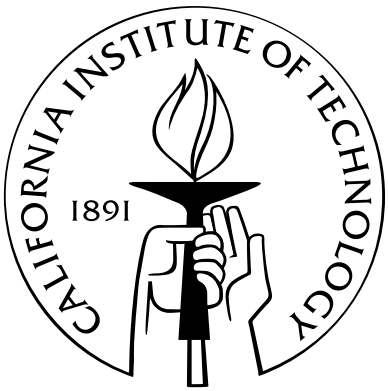
Thesis by

Brian R. Wolfe

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy



California Institute of Technology

Pasadena, California

2014

(Defended May 6, 2014)

Acknowledgements

I would like to thank my advisor, Niles Pierce. Under his guidance I have learned how to approach and study problems in depth in order to understand their underlying kernels of truth. His principled, focused approach to every aspect of his work has been extremely instructive. I have also been fortunate to work with Erik Winfree and his research group. These interactions have helped direct my research to create practical design tools for the nucleic acid engineering community. I would also like to thank the other two members of my committee, Tom Miller and Richard Murray, for their insight and advice.

Within the Pierce Lab, I worked with many skilled researchers. Joe Zadeh and I worked together on multi-complex design algorithms during my first year in the lab. His work on efficient ensemble defect optimization elucidated ingredients that formed the basis for the design algorithms presented in this thesis. I have also had the opportunity to work with Justin Bois on several projects. His understanding of thermodynamics and statistical mechanics have been invaluable. In addition, his passion for discovery and thirst for knowledge are an inspiration for those he works with. The remaining members of the Pierce Lab have made my work both feasible and useful. My work would not have been possible without the partition function and coarse-graining algorithms developed before my arrival, and the experimentalists in the Pierce Lab have been incredible to work with and have been willing to use, find bugs in, and suggest changes to many of the tools I developed.

I would also like to thank my parents, who have always been patient and helpful teachers. They inspired me through their work and through their knowledge. Finally, I would like to thank my wife Helen, who is a partner in both love and intellectual adventures.

Abstract

Nucleic acids are a useful substrate for engineering at the molecular level. Designing the detailed energetics and kinetics of interactions between nucleic acid strands remains a challenge. Building on previous algorithms to characterize the ensemble of dilute solutions of nucleic acids, we present a design algorithm that allows optimization of structural features and binding energetics of a test tube of interacting nucleic acid strands. We extend this formulation to handle multiple thermodynamic states and combinatorial constraints to allow optimization of pathways of interacting nucleic acids. In both design strategies, low-cost estimates to thermodynamic properties are calculated using hierarchical ensemble decomposition and test tube ensemble focusing. These algorithms are tested on randomized test sets and on example pathways drawn from the molecular programming literature. To analyze the kinetic properties of designed sequences, we describe algorithms to identify dominant species and kinetic rates using coarse-graining at the scale of a *small box* containing several strands or a *large box* containing a dilute solution of strands.

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

DNA and RNA are information storage, transmission, and processing materials for biological systems. Recently, researchers have used nucleic acids as a nanoscale engineering material, constructing novel structures, devices, and systems. The initial nucleic acid designs were static assemblies; the earliest was a sixty-four nucleotide long four-arm junction [67]. Structural DNA nanotechnology has advanced tremendously since then [57, 66]. Researchers today produce DNA origami, brick, and crossover structures consisting of thousands of strands [19, 39, 59]. In the meantime, dynamic DNA nanotechnology has also proliferated, and a wide range of behaviors has been demonstrated [85]. Researchers have created boolean logic circuits [65], spatiotemporal oscillators [2, 55], conditional and catalytic structure assembly [15, 79], DNA walkers [6, 45, 77], and many other assemblies with triggerable state changes [6, 31, 80]. Additionally, practical tools have been built for detecting mRNA expression, proteins, and other analytes [7, 12, 11]. Technologies to conditionally interact with mRNA, siRNA, and biological nucleic acid pathways point to even more opportunities to study, treat, and engineer biological systems using rationally designed nucleic acid systems [33, 36].

These diverse nucleic acid systems are feasible to design because nucleic acids are typically dominated by the energy of local interactions. The energetics of folding a small nucleic acid complex are dominated by the contributions of base pairing of opposing nucleotides in a double helix and base stacking of adjacent base pairs. These simple energetics permit the use of empirical energy models and design tools based on nearest-neighbor interactions. Chapter 2 of this thesis describes a few rationally designed nucleic acid systems, the nearest neighbor model, and previous algorithms relating to this thesis.

Previous thermodynamic nucleic acid design tools address the design of single strands or complexes in isolation. These algorithms do not consider the possibility of off-target complexes forming in solution. Chapter 3 describes a test tube design algorithm that addresses this deficiency, enabling efficient optimization of the equilibrium base-pairing properties of a test tube of interacting nucleic acid strands. This chapter is heavily based on work that has been submitted for publication.

Optimization of a single state is useful for designing static assemblies, but optimization of dy-

namic nucleic acid systems requires consideration of multiple states. Many approaches are currently used to do this, but these are typically limited to a subset of the thermodynamic model, and often rely primarily on combinatorial approaches to design against undesired interactions. In Chapter 4, we describe a multistate thermodynamic design algorithm that optimizes multiple dilute solutions and target structures while satisfying combinatorial sequence constraints. By designing initial, intermediate, and final states of reaction pathways, nucleic acid engineers can design sequences for dynamic nucleic acid systems. By allowing combinatorial sequence constraints, engineers can restrict the allowed sequence space and enforce interactions not included in the thermodynamic model. The work in this chapter is being prepared for publication.

Understanding the dynamic behavior of the resulting sequence designs can be important to predicting their effectiveness. There has been significant recent work simulating nucleic acids at various levels of detail. Chapter 5 describes a coarse-graining algorithm to find a high level description for the kinetics of a small number of nucleic acid strands, and its extension to find the mass-action kinetics for a test tube of interacting nucleic acid strands. These algorithms are modifications and extensions of a previous coarse-graining algorithm developed by Jonathan Othmer [51].

Chapter 6 summarizes the impact of this work.

The Appendices contain details and subtleties that are not crucial to the main thesis. Appendix A contains the pseudocode of several algorithms that were useful as references for understanding the algorithms in this thesis, for writing certain sections of the code, or for generating test sets. Appendix B contains parameter sensitivity studies and further characterization of the test tube design algorithm. Appendix C contains a description of the NUPACK design language and descriptions of material in the supplementary archive. Appendix D contains a description of material in the supplementary archive and a brief discussion of error bounds that may be helpful for future development.

A supplementary archive contains input files, structures, and test set generation scripts that may be used to replicate the studies in this thesis.

Chapter 2

Model and background

Here, we describe the types of systems we aim to design and analyze, the underlying free energy model and kinetic model, and previous work to computationally design nucleic acids and coarse-grain their kinetics. Section 2.1 describes basic terminology and properties of nucleic acids. Section 2.2 describes some rationally designed nucleic acid systems that are used as examples in this work. Section 2.3 describes the empirical nearest neighbor thermodynamic model and kinetic model for nucleic acids. Section 2.4 describes previous work in nucleic acid sequence design. Section 2.5 describes previous work on coarse-graining the kinetics of nucleic acid secondary structures.

2.1 Nucleic acids

DNA and RNA are poly-nucleotides. Each nucleotide consists of a negatively charged phosphate, a sugar, and a nucleobase that can differ between nucleotides. Each molecule consists of a specific ordering of the four bases: A, C, G, and U for RNA, where T replaces U for DNA. The phosphate groups join the $5'$ and $3'$ carbons of adjacent sugars to form the backbone of the molecule. The ends of the molecule are labeled $5'$ and $3'$, corresponding to the position of the carbon closest to the end, e.g., sequence $5'$–`GGAUGUA`–$3'$. A double helix of nucleic acids consists of two anti-parallel strands or strand segments. The strands are held together by a combination of hydrogen bonding between opposing bases that form base pairs and stacking between adjacent bases and base pairs. Base pairs are typically either Watson-Crick pairs, [`A`·`U`] and [`G`·`C`], or wobble pairs, [`G`·`U`]. Base pairing can occur both between strands and between subsequences on a single strand. The complement of a nucleotide is the nucleotide that forms a Watson-Crick pair with it. The complement of a sequence is the sequence in which all opposing nucleotides are complementary when aligned antiparallel with the original. A secondary structure for an ordered list of strands is a set of base pairs between the constituent nucleotides.

For example, the strand $5'$–`GCAUG`–$3'$ is the complement of $5'$–`CAUGC`–$3'$. These strands can form a perfect duplex. Also, the first five bases of the nucleic acid strand $5'$–`GCAUGAAAACAUGC`–$3'$ are

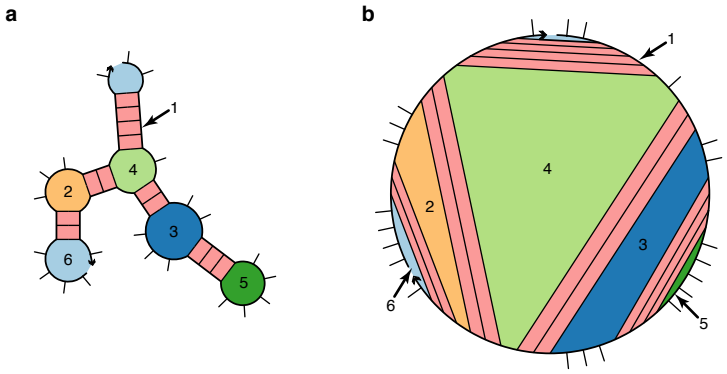


Figure 2.1: a) A secondary structure. b) A polymer graph representation of the same structure. Loops are colored according to their type: 1) stacked loop, 2) bulge loop, 3) interior loop, 4) multiloop, 5) hairpin loop, 6) exterior loop.

complementary to the last five bases. This sequence can form a hairpin, where the beginning and end of the structure come together, leaving a small loop. Secondary structures can be drawn as a backbone connected by base pairs, as shown in Figure 2.1a.

It is also helpful to be able to specify degenerate sequences, where each position matches a subset of allowed nucleotides. These sequence patterns can be specified using IUPAC notation, as shown in Table 2.1. For example, sequence pattern 5′–`SSAANN`–3′ matches 64 possible RNA sequences, including 5′–`GCAAGU`–3′.

In nature, the specificity provided by complementarity is used during replication of DNA, transcription of RNA, translation of RNA into proteins, and post-transcriptional regulation. Taking advantage of this specificity, researchers have been able to rationally design molecules, systems, and devices out of nucleic acids [57, 86].

2.2 Rationally designed nucleic acid systems

Many rationally designed nucleic acid systems use toehold mediated strand-displacement [80, 86]. In toehold mediated strand displacement, a duplex ends with a short unstructured region called a toehold. A new strand, complementary to the toehold and the continuation of that strand in the duplex, can bind to the toehold and displace the partner strand in a random walk [80].

This basic mechanism has been used to implement many behaviors, including AND gates (Figure 2.2), conditional assembly of long polymers (Figure 2.3), catalytic assembly of 3-armed junctions (Figure 2.4), cooperative gates (Figure 2.5), and conditional assembly of biologically relevant complexes (Figure 2.6). The mechanism for each system is described in the figure captions. We will use these design types as examples in the multistate design and kinetic coarse-graining chapters of this thesis.

Symbol	Possible nucleotide(s)
A	A
C	C
G	G
T	T
U	U
R	A or G
Y	C, T, or U
M	A or C
K	G, T, or U
W	A, T, or U
S	C or G
B	C, G, T, or U
D	A, G, T, or U
H	A, C, T, or U
V	A, C, or G
N	A, C, G, T or U

Table 2.1: IUPAC notation for degenerate nucleotides. The symbol on the left represents any of the possible nucleotides listed on the right.

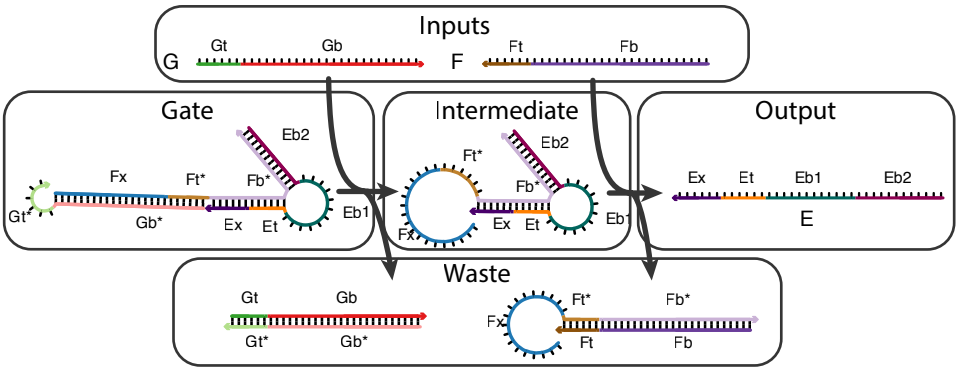


Figure 2.2: DNA AND gate, from Seelig et al. [65]. In this mechanism, the gate reacts with inputs G and F via toehold mediated strand displacement to produce output E and two waste molecules. Input G binds to the gate via domain Gt and performs a branch migration; simultaneously creating the first waste duplex while exposing toehold Ft. Input F binds to the newly exposed toehold and performs a second branch migration across domain Fb, producing a waste duplex and releasing output molecule E.

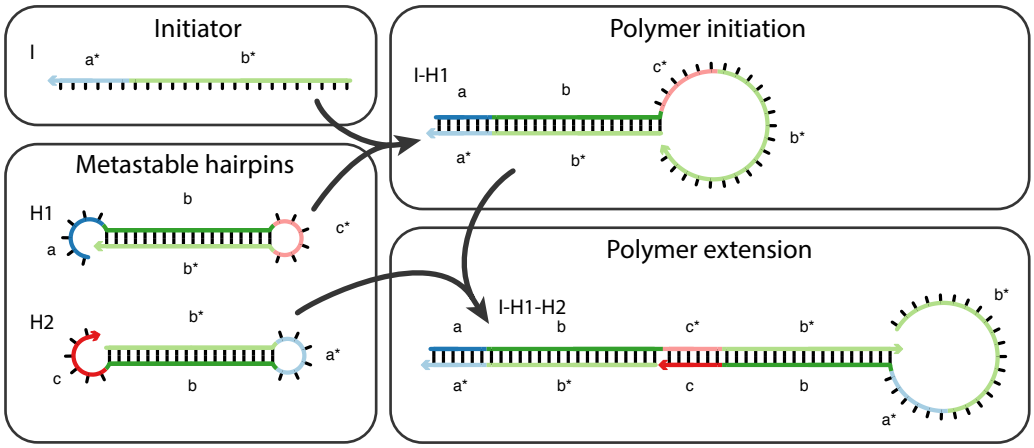


Figure 2.3: Hybridization chain reaction (HCR) mechanism, from Dirks et al. [17]. In this mechanism, an initiator molecule, I, starts a polymerization of hairpins H1 and H2. Initially, I binds to toehold a of H1, performs a branch migration to open the hairpin, and exposes sequestered toehold c^*. Subsequently, H2 can bind to the newly exposed region and perform a symmetric addition step. This exposes a tail that is identical to initiator I, allowing this process to repeat. In the absence of I, the hairpins should be metastable and polymerization should be slow.

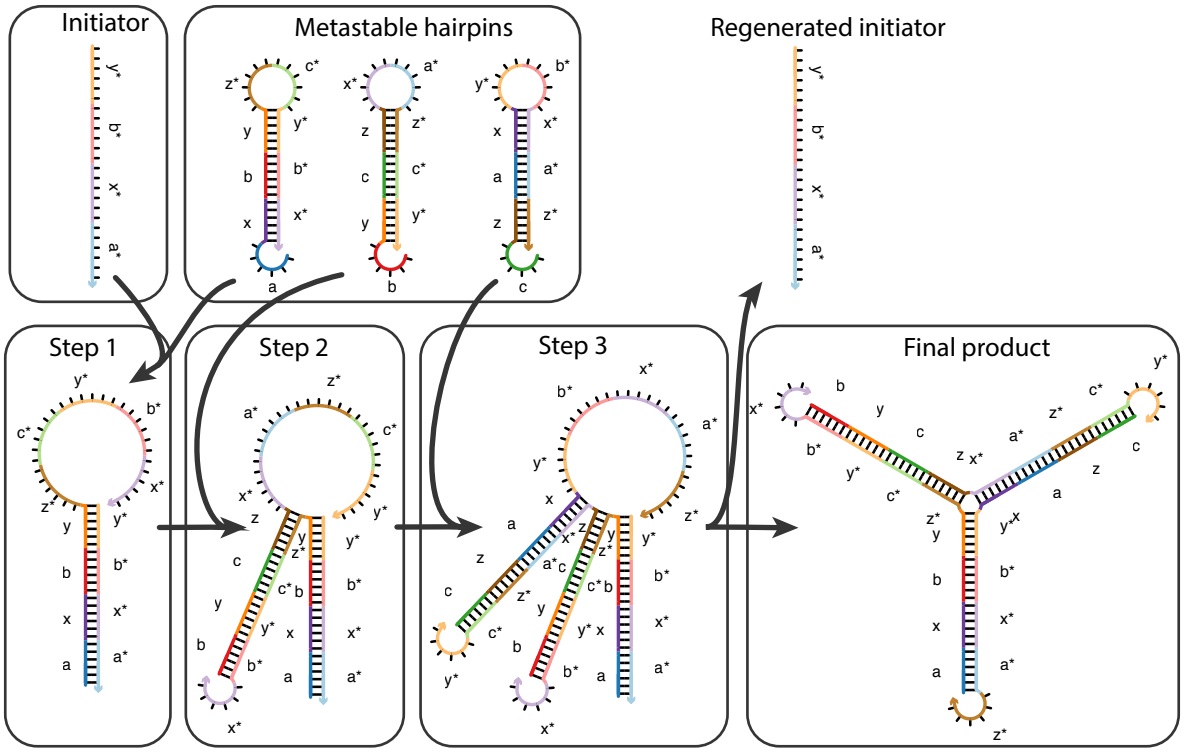


Figure 2.4: Three-arm junction mechanism, from Yin et al. [79]. In this mechanism, an initiator molecule catalyzes the formation of three-arm junctions. Starting with the unstructured initiator, each hairpin adds to the free end of the complex, performs a branch migration, and exposes a new unstructured tail. After the third hairpin is added, the initiator, which is identical to the first four domains of the now-free tail of the third hairpin, is regenerated via a final branch-migration step.

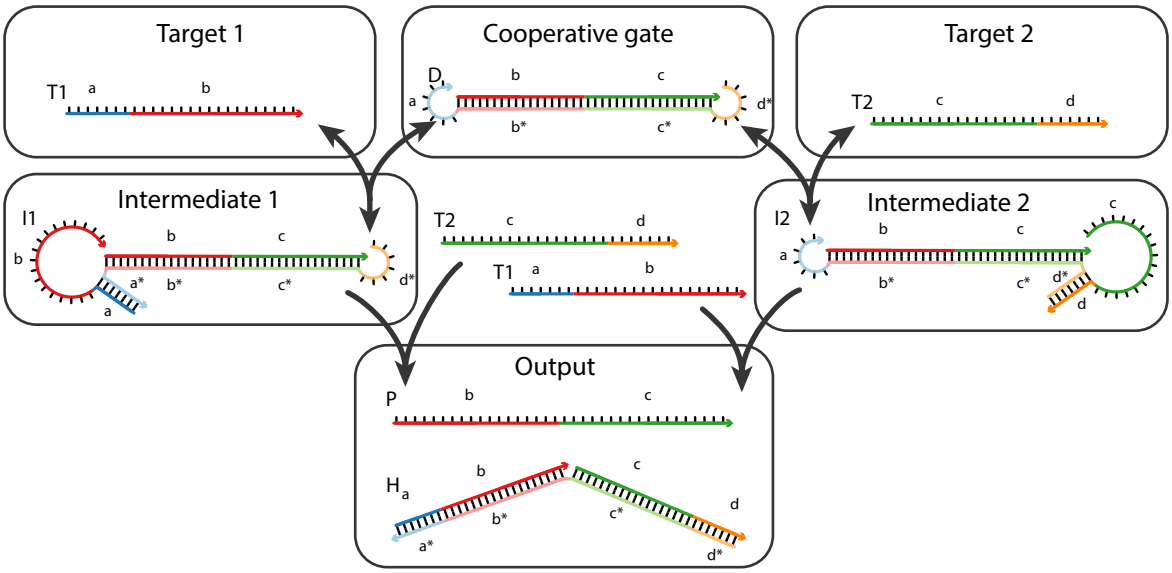


Figure 2.5: Cooperative AND gate mechanism, from Zhang [84]. In this mechanism, two input molecules, T1 and T2, cooperatively bind to gate D to displace output strand P. Starting with gate D, T1 can bind to toehold a^* or T2 can bind to toehold d^*. In the absence of the other input, any branch migration will be unproductive since many base pairs will remain between the two gate strands, and the input will eventually dissociate. When both inputs bind to the same molecule, however, they can both branch migrate and cooperatively release output P.

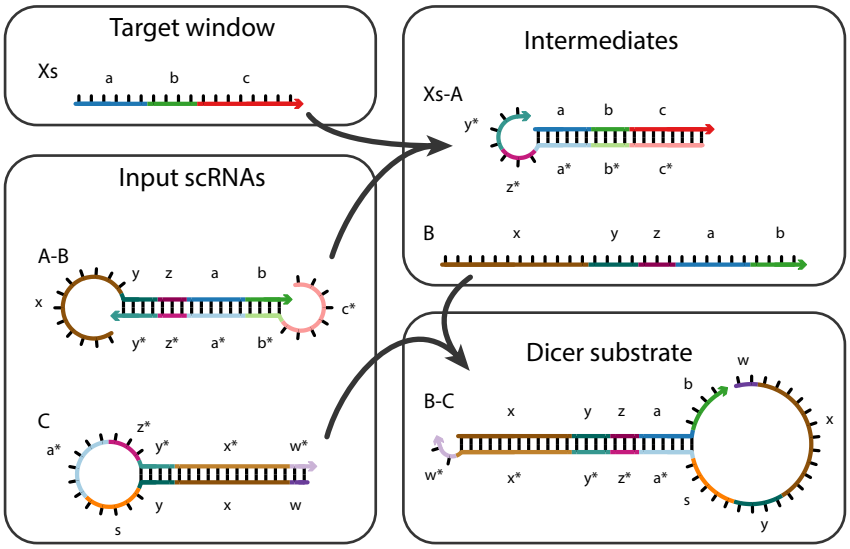


Figure 2.6: Conditional dicer substrate mechanism, from Hochrein et al. [33]. In this mechanism, a sequence from an mRNA conditionally produces a substrate for the dicer enzyme that targets an independent mRNA sequence. Starting with a short window from the target mRNA, Xs, input A-B binds, producing waste dimer Xs-A and single-stranded molecule B. This molecule is then free to perform loop invasion and branch migration on molecule C, producing dicer substrate B-C. Note that the input sequence a, b, c, is independent from the first 19 nucleotides of the dicer substrate, x, y, z. These are used to specify the downstream target.

2.3 Physical model

The sequence, ϕ, of one or more interacting RNA strands is specified as a list of bases $\phi^a = \{A, C, G, U\}$, for $a = 1, \ldots, |\phi|$; `T` replaces `U` for DNA. A secondary structure, s, of one or more interacting nucleic acid strands is defined by the set of base pairs $[a \cdot b]$. A polymer graph representation of a secondary structure is constructed by ordering the strands around a circle, drawing the backbones in succession from $5'$ to $3'$ around the circumference with a nick between each strand, and drawing straight lines connecting paired bases. A secondary structure is unpseudoknotted if and only if there exists a strand ordering for which the polymer graph has no crossing lines. A secondary structure is connected if and only if every subset of the strands can be reached from every other by traversing only the backbones within strands and base pairs between strands. The secondary structure in Figure 2.1a is accompanied by its corresponding polymer graph representation in panel b.

A complex of interacting strands with strand ordering π has structural ensemble $\Gamma(\pi)$, containing all connected, unpseudoknotted secondary structures. For sequence ϕ and secondary structure $s \in \Gamma$, the free energy $\Delta G(\phi, s)$, is calculated using nearest-neighbor empirical parameters for RNA at 1 M Na^+ [48, 47] or for DNA in user-specified Na^+ and Mg^{2+} [54, 56, 62, 63].

A particular secondary structure can be decomposed into its constitutive loops; each loop is bounded by the backbone and a set of zero or more base pairs. Loops are categorized by the number of bounding base pairs and by the presence or absence of nicks. All loops that contains a nick are exterior loops. Hairpin loops are bounded by exactly one base pair. Interior loops are bounded by exactly two base pairs. Multiloops are bounded by at least three base pairs. Interior loops are further divided into stacked loops (between two adjacent base pairs), bulge loops (between two base pairs with at least one nucleotide interceding on exactly one side) and other interior loops. The nearest neighbor model assigns a free energy to each loop based on its type and constitutive nucleotides. The loops in the secondary structure depicted in Figure 2.1 are colored according to their type.

In addition to the loop free energies, a complex containing L strands has a free energy cost associating those strands of $(L-1)\Delta G^{\text{assoc}}$ [14]. If the complex contains indistinguishable strands, the structure may be symmetric. For a structure, s, with rotational symmetry $R(\phi, s)$, the free energy must be adjusted by a symmetry correction $k_B T \log[R(\phi, s)]$, accounting for the reduction in the number of distinguishable states. The free energy, $\Delta G(\phi, s)$, of sequence ϕ folded into secondary structure s is estimated as the sum over loop free energies, the association energy, and the rotational correction [14]:

$$\Delta G(\phi, s) = k_B T \log R(\phi, s) + (L(\phi) - 1)\Delta G^{\text{assoc}} + \sum_{\text{loop} \in s} \Delta G(\phi_{\text{loop}}). \tag{2.1}$$

2.3.1 Analyzing equilibrium properties

Let Ψ^0 denote the set of strand species that interact in a test tube to form the set of complex species Ψ. For complex $j \in \Psi$, with sequence ϕ_j and structural ensemble Γ_j, the partition function,

$$Q(\phi_j) = \sum_{s \in \Gamma_j} \exp[-\Delta G(\phi_j, s)/k_B T],$$

can be used to calculate the equilibrium probability of any secondary structure $s \in \Gamma_j$:

$$p(\phi_j, s) = \exp[-\Delta G(\phi_j, s)/k_B T]/Q(\phi_j).$$

The secondary structure with the highest probability in Γ_j is the minimum free energy structure $s^{\mathrm{MFE}}(\phi_j)$, satisfying

$$s^{\mathrm{MFE}}(\phi_j) = \underset{s \in \Gamma_j}{\operatorname{argmin}} \Delta G(\phi_j, s).$$

Here, k_B is the Boltzmann constant and T is the temperature. The equilibrium base pairing properties of complex j are characterized by the base pairing probability matrix $P(\phi_j)$, with entries $P^{a,b}(\phi_j) \in [0, 1]$ corresponding to the probability,

$$P^{a,b}(\phi_j) = \sum_{s \in \Gamma_j} p(\phi_j, s) S^{a,b}(s),$$

that base pair $[a \cdot b]$ forms at equilibrium within ensemble Γ_j. Here, $S(s)$ is the structure matrix of structure s. $S^{a,b}(s) = 1$ if structure s contains base pair $[a \cdot b]$, and is zero otherwise. $S^{a,|s|+1}(s) = 1$ if base a is unpaired and is zero otherwise. Hence, the entry $P^{a,|\phi_j|+1}(\phi_j) \in [0, 1]$ denotes the equilibrium probability that base a is unpaired over ensemble Γ_j, and the row sums of the augmented $S(s)$ and $P(\phi)$ matrices are unity. Let $Q_\Psi \equiv Q_j \ \forall j \in \Psi$ denote the set of partition functions of all complexes in a test tube. The set of equilibrium concentrations, x_Ψ, (specified as mole fractions) are the unique solutions to the strictly convex optimization problem [14]:

$$\underset{x_\Psi}{\operatorname{argmin}} \sum_{j \in \Psi} x_j (\log x_j - \log Q_j - 1) \tag{2.2a}$$

$$\text{subject to } A_{i,j} x_j = x_i^0 \ \ \forall i \in \Psi^0., \tag{2.2b}$$

where the constraints impose conservation of mass. A is the stoichiometry matrix with entries $A_{i,j}$ corresponding to the number of strands of type i in complex j, and x_i^0 is the total concentration of strand i introduced to the test tube.

To analyze the equilibrium base pairing properties of a test tube, the partition function, $Q(\phi_j)$,

and equilibrium pair probability matrix, $P(\phi_j)$, must be calculated for each complex, $j \in \Psi$, using $\Theta(|\phi_j|^3)$ dynamic programs. The equilibrium concentrations, x_Ψ, are calculated by solving the convex optimization problem (2.2) using an efficient trust region method at a cost that is typically negligible by comparison [14]. The overall time complexity to analyze the test tube is then $O(|\Psi||\phi|^3_{\max})$, where $|\phi|_{\max}$ is the size of the largest complex.

In specifying an analysis problem, a convenient and powerful approach to define Ψ is to include all complexes up to $L_{\max}$ strands. For a test tube containing the set of strands, Ψ^0, the total number of complexes that can form up to size $L_{\max}$ is

$$|\Psi| = \sum_{L=1}^{L_{\max}} \sum_{l=1}^{L} \frac{|\Psi^0|^{\gcd(l,L)}}{L} \tag{2.3}$$

so the overall time complexity to analyze the test tube is $O(|\Psi^0|^L_{\max}|\phi|^3_{\max}/L_{\max})$ [14].

The set of possible complexes is equivalent to the set of all possible necklaces over alphabet Ψ^0 from length 1 to $L_{\max}$. The FKM algorithm [60] can be used to efficiently enumerate the set of complexes (see Appendix Section A.1).

2.3.2 Analyzing kinetic properties

Kinetic properties of nucleic acids have been studied at many levels of abstraction, including atomic detail molecular dynamics [10, 42], coarse-grained molecular dynamics [52, 76], and secondary structure kinetics [21, 64]. We choose to use the Multistrand kinetic model [64]; here, each secondary structure is a state in a continuous time Markov chain, and states are connected by the breaking or forming of single base pairs. By enforcing connectivity and detailed balance with respect to the free energies assigned by the thermodynamic model, the probability of being in a particular state will converge to the equilibrium probabilities as calculated in the previous section [64, 75].

To consider the kinetics of a solution of interacting nucleic acids, we first define the secondary structure of a small box, ω, as a set of complexes and their corresponding secondary structures. The ensemble of all unpseudoknotted secondary structures of the box is Ω^*. The free energy of each complex can be determined from (2.1). The free energy of the box additionally accounts for the free energy of mixing, which, assuming the nucleic acids are vastly outnumbered by water, can be written [64]:

$$\Delta G^{\text{box}}(\omega_i) = (|\Psi_0| - L(\omega_i))k_B T \log\left(\frac{V}{V_0}\right) + \sum_{j \in \omega_i} \Delta G(s_j, \phi_j). \tag{2.4}$$

Here, $L(\omega_i)$ is the number of complexes in secondary structure ω_i. Using this, we can define secondary structure matrices, $S(\omega)$, and pair probability matrices, $P(\phi)$, analogously to the single-complex definitions.

Each small box secondary structure, ω_i, is a state in the secondary structure kinetic model.

Since the kinetic model assumes the Markov property, we can write down the corresponding master equation describing the time-varying probabilities for each secondary structure ω_j at time t

$$\frac{d}{dt}\bar{p}_j(t) = \sum_{i \in \Omega^*} \bar{r}_{i \to j}\bar{p}_i(t) - \bar{r}_{j \to i}\bar{p}_j(t) \tag{2.5}$$

Collecting the rates into the rate matrix $R(\phi)$, where $R_{i,j}(\phi) = \bar{r}_{j \to i}$, we can write this as

$$\frac{d}{dt}\bar{p}(t) = R(\phi)\bar{p}(t). \tag{2.6}$$

Given a vector of initial secondary structure probabilities, $\bar{p}(0)$, this has the solution

$$\bar{p}(t) = e^{R(\phi)t}\bar{p}(0). \tag{2.7}$$

To find reaction rates consistent with the nearest neighbor thermodynamic model, we enforce detailed balance between secondary structures, constraining the ratio of forward and reverse rates so that no flux occurs at equilibrium, i.e.,

$$\bar{r}_{i \to j}\bar{p}(\omega_i, \phi) = \bar{r}_{j \to i}\bar{p}(\omega_j, \phi). \tag{2.8}$$

This defines each ratio, but the scaling of each pair of rates still needs to be determined. In practice, the kinetic rates are set either via the Metropolis rule [50],

$$r_{i \to j} = \begin{cases} k_{\text{uni}} \exp[-(\Delta G_j - \Delta G_i)/k_B T)] & : \Delta G_j \geq \Delta G_i \\ k_{\text{uni}} & : \Delta G_j < \Delta G_i \end{cases}, \tag{2.9}$$

or the Kawasaki rule [38],

$$r_{i \to j} = k_{\text{uni}} \exp[-(\Delta G_j - \Delta G_i)/2k_B T)]. \tag{2.10}$$

As in all Markov processes, self-transitions are governed by

$$r_{i \to i} = -\sum_{j \neq i, j \in \Gamma} r_{i \to j} \tag{2.11}$$

to conserve probability. Both rate rules enforce detailed balance and every state is reachable from every other state by single base pair changes (for any given starting and ending secondary structure, consider that all base pairs from the starting secondary structure can be removed sequentially and then base pairs corresponding to the final state can be added sequentially, so every other state is reachable). These properties are sufficient to guarantee that the Markov chain is irreducible and the

probability of sampling a particular state converges to the equilibrium probability of the state as described by the thermodynamic model [75].

The master equation can be simulated directly for sufficiently small systems. However, the number of structures grows exponentially in the length of the nucleic acid strand, which necessitates the use of the Monte Carlo simulators such as Multistrand [64].

These tools provide mechanisms to measure mean first-passage times, and so they can be used to estimate kinetic rates. The simulations for toe-hold mediated , however, have not yet consistently predicted experimental reaction rates. By using finer grained simulations, Srinivas et al. [72] recently discovered modifications to the thermodynamic and kinetic model that reduce some of this error (it was shown that the rate laws will need to be adjusted to accurately match experimental results).

2.4 Previous design algorithms

A wide range of design algorithms have been developed for the the design of single complexes and for the design of pathways of interacting strands. Here, we provide an overview of these methods.

2.4.1 Single-complex design

Single complex design has been approached in two ways. The first approach is to view design as an optimization process, using an adaptive walk to find local minima according to some measure of the defect for each candidate sequence. The second approach is to view design as a constraint satisfaction problem, where combinatorial and thermodynamic constraints are used to generate a set of feasible sequences. Efficient design using either approach takes advantage of the locality of the energy function to decompose the design problem.

Thermodynamic optimization approaches attempt to optimize a nucleic acid sequence according to a given objective function. The following objective functions have been used:

- The *MFE defect*,

$$n(\phi, s) = |s| - \sum_{\substack{1 \le a \le |\phi| \\ 1 \le b \le |\phi|+1}} S(s^{\mathrm{MFE}}(\phi)) S(s), \tag{2.12}$$

  the number of nucleotides paired differently between the target structure and the minimum free energy structure [4, 9, 44].

- The *probability defect*,

$$\chi(\phi, s) = 1 - p(s, \phi), \tag{2.13}$$

  the cumulative equilibrium probability of all structures except the target structure.

- The *ensemble defect*,

$$n(\phi, s) = |s| - \sum_{\substack{1 \le a \le |\phi| \\ 1 \le b \le |\phi|+1}} P^{a,b}(\phi) S(s), \tag{2.14}$$

  the average number of incorrectly paired nucleotides at equilibrium [15, 83].

For the chosen objective, an algorithm typically performs a local search, mutating bases or base pairs. More recent approaches have applied other metaheuristics, including genetic algorithms and global sampling [43, 73] to the MFE optimization problem.

To allow efficient design, many of these algorithms take advantage of the locality of the free energies to implement a divide-and-conquer approach to sequence design. The target structure is hierarchically decomposed based on dominant features (multiloops or helices), producing a tree of smaller substructures. The leaves of this tree are designed independently and the resulting sequences are merged up the tree, recursively reconstructing the sequence of the root. If the merged sequences exhibit defects not present in the children, a perturbation is applied to the sequences and optimization is reattempted using the same decomposition. This decomposition approach has been used for both MFE structure optimization and ensemble defect optimization. Using these approaches, both MFE design and ensemble defect optimization can exhibit asymptotic optimality. Due to the $\Theta(N^3)$ cost of evaluating the MFE structure and ensemble defect, both design types must satisfy an optimality bound: the minimal cost of design is at least 4/3 the cost of evaluating the defect if at least one mutation is made. As target structures that allow decomposition get longer, the typical design cost empirically approaches this 4/3 optimality bound [83].

Most single-complex design algorithms focus on MFE defect optimization (also called inverse folding). The weakness of this approach is its assumption that the MFE structure dominates the base pairing properties of the ensemble. While it is the most probable structure, the base pairs in the MFE structure can have arbitrarily low probability. By assuming that the MFE structure dominates the ensemble, the algorithm may ignore defects that are obvious from the base pairing probabilities. The ensemble defect, on the other hand, captures this type of design flaw. The advantages of using the more physically meaningful ensemble defect instead of the other objectives are discussed in detail by Dirks and Zadeh [15, 83].

Constraint-programming-based design of single complexes has recently been developed. This takes advantage of progress in development of high-performance constraint-programming algorithms and enumerates the set of sequences that satisfy combinatorial and thermodynamic constraints [25]. These approaches allow one to directly enumerate all sequences that, for example, satisfy complementarity for two different folds and have an MFE of exactly the target structure. Constraint programming is the only approach that can determine that no sequence exists with an MFE structure

matching the target structure. This approach also allows elegant specification of combinatorial and other constraints using the same constraint satisfaction framework.

2.4.2 Pathway design

The design of pathways of interacting nucleic acids is typically performed much differently. Many design approaches in the molecular programming community use little to no thermodynamic modeling during design. These design approaches typically rely on pure combinatorial approaches, design heuristics based on past experience, or design-specific thermodynamic approaches that optimize the thermodynamics of particular structural features [20, 70, 85].

Pure combinatorial algorithms attempt to optimize against off-target binding by minimizing ad-hoc measures of binding like sequence symmetry. These tools scale well to a large number of sequences and the simple energetics of nucleic acids have permitted their use. By ignoring the empirical model completely, however, these approaches cannot be predictive of failure modes and cannot motivate or take advantage of improvements in the free energy model. One heuristic designer was created by David Zhang and used to design many experimental systems [85].

To address the problems of designing with respect to the known thermodynamic parameters, specialty algorithms have been developed to design sequence domains using a subset of the thermodynamic model. For example, the algorithm in Evans et al. does a complete enumeration of all pairs of sticky ends [20] to find a subset with minimal off-target binding and nearly iso-energetic on-target binding. Shortreed et al. [70] used a combination of thermodynamic and combinatorial metrics to design a similar set of orthogonal subsequences.

In contrast to the combinatorial and the domain-based optimization algorithms, a multi-structure thermodynamic design algorithm was presented in Joseph Zadeh's thesis [81]. This algorithm optimizes the ensemble defect of multiple complexes simultaneously, subject to several basic sequence constraint types. This algorithm succeeds at rigorously using the free energy model to optimize checkpoint structures in reaction pathways, but cannot optimize against off-target complexes, limiting its utility.

The algorithms presented in Chapters 3 and 4 introduce test tube design and multistate design for a set of test tubes. Test tube design allows specification of a set of 'on-target complexes', each with a target structure and target concentration, and a set of 'off-target' complexes, each with vanishing target concentration. We extend this with multistate design, elegantly capturing thermodynamic design of nucleic acid reaction pathways in the context of a dilute solution, and rigorously optimizing against off-target complexes. Additionally, we generalize the constraint handling of previous thermodynamic optimization approaches, allowing engineers to use a wide variety of combinatorial constraints during the optimization process.

2.5 Previous coarse-graining algorithms

There are many tools that can be used to compute the features of a folding landscape for single strands of RNA. For small sequences, a barrier tree representation of the secondary structure landscape can be constructed, caputuring all minima and the saddle points between them [23, 78]. If starting and ending secondary structures are specified, kinetics between them can be projected to a two dimensional distance matrix, and the kinetics through this reduced dimension space can be estimated, providing reasonable estimates of the full simulation [68]. One can also perform a semi-greedy search for low energy paths between two structures, as described in [18].

Another approach constructs a set of local minima by repeatedly sampling secondary structures and relaxing them to their nearest minimum [41]. These local minima can be used as the start and end points for any of the path search algorithms described above, and kinetics can then be derived from the resulting barrier heights. These methods are practical tools for exploring the folding landscape for single strands.

For multiple strands, macroscopic kinetic properties have been explored using Monte Carlo simulations between starting and ending states, and between sampled first contact states and a set of ending states [64].

In Chapter 5, we present a trajectory-based coarse-graining for a small box containing a few nucleic acid strands and for a test tube containing a large number of strands, estimating the master equation and mass-action kinetics of the respective systems.

Chapter 3

Nucleic acid sequence design for dilute solutions of interacting strands

In this chapter, we describe an algorithm for designing the equilibrium base pairing properties of a test tube of interacting nucleic acid strands. A target test tube is specified as a set of desired 'on-target' complexes, each with a target secondary structure and target concentration, and a set of undesired 'off-target' complexes, each with vanishing target concentration. Sequence design is performed by optimizing the test tube ensemble defect, corresponding to the concentration of incorrectly paired nucleotides at equilibrium evaluated over the ensemble of the test tube. To reduce the computational cost of accepting or rejecting mutations to a random initial sequence, on-target structures are each decomposed into a tree of substructures, yielding a forest of decomposition trees. The influence of each candidate mutation on the test tube ensemble defect is estimated using nodal defect contributions calculated efficiently over the leaf subensembles. As optimized subsequences are merged moving toward the root level of the forest, any decomposition defects are eliminated via ensemble redecomposition and sequence reoptimization. After successfully merging subsequences to the root level, the exact test tube ensemble defect is calculated for the first time, explicitly checking for the effect of the previously neglected off-target complexes. Any off-target complexes that form at appreciable concentrations are hierarchically decomposed, added to the decomposition forest, and actively destabilized during subsequent forest reoptimization. For target test tubes representative of design challenges in the molecular programming and synthetic biology communities, RNA sequence design at 37 °C typically succeeds in achieving a normalized test tube ensemble defect $\leq 1\%$ at a design cost within an order of magnitude of the cost of a single evaluation of sequence quality.

3.1 Introduction

The programmable chemistry of nucleic acid base pairing serves as a versatile medium for the rational design of self-assembling molecular structures, devices, and systems [57, 84]. To assist in these engineering efforts, analysis algorithms have been developed to enable calculation of the equilibrium base pairing properties of a dilute solution of interacting nucleic acid strands (e.g., a test tube), yielding predictions for the equilibrium concentration and base pairing probabilities for an arbitrary number of complex species that form from an arbitrary number of strand species [14, 82]. Of course, in an engineering setting, sequence analysis must be preceded by sequence design. However, no analogous sequence design algorithm exists for engineering the equilibrium base pairing properties of a test tube of interacting nucleic acid strands.

To date, considerable effort has been invested in addressing the crucial subsidiary challenge of designing the equilibrium base pairing properties of a single complex of (one or more) interacting nucleic acid strands [1, 3, 5, 8, 9, 15, 16, 22, 24, 34, 43, 46, 49, 58, 71, 73, 74, 83]. For *complex design*, the user specifies a target secondary structure for the complex; neither the concentration of the complex, nor the concentrations of other undesired complexes are considered. As a result, sequences that are successfully optimized to stabilize a target secondary structure in the context of a complex may nonetheless fail to ensure that this complex forms at appreciable concentration when the strands are introduced into a test tube (see Figure 3.1). To address this major conceptual and practical shortcoming, this chapter formulates nucleic acid sequence design in the context of a test tube of interacting nucleic acid strands at equilibrium. For *test tube design*, the user specifies: 1) a set of desired 'on-target' complexes, each with a target secondary structure and target concentration, and 2) a set of undesired 'off-target' complexes, each with vanishing target concentration.

We have previously shown that complex design can be formulated as an optimization problem based on a physically meaningful objective function, the complex ensemble defect [82, 83]. For a candidate sequence and target secondary structure, the complex ensemble defect is the average number of incorrectly paired nucleotides at equilibrium evaluated over the ensemble of the complex [15]. Here, to provide a physically meaningful objective function for test tube design, we derive the test tube ensemble defect, corresponding to the concentration of incorrectly paired nucleotides at equilibrium evaluated over the ensemble of the test tube. To efficiently optimize the test tube ensemble defect, we build on hierarchical sequence optimization concepts previously developed for complex design [1, 3, 9, 34, 83] by deriving a hierarchical decomposition of the test tube ensemble.

3.1.1 Test tube design problem specification

A *test tube design* problem is specified as a *target test tube* containing a set of desired *on-target complexes*, Ψ_{on}, and a set of undesired *off-target complexes*, Ψ_{off}. The set of complexes in the test

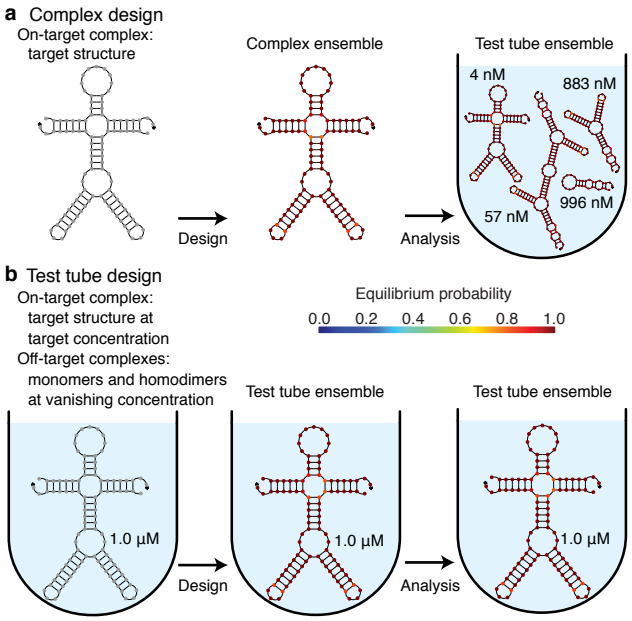


Figure 3.1: Complex design versus test tube design. (a) Complex design. Sequence design formulated in the context of a complex (left) ensures that at equilibrium the target structure dominates the structural ensemble of the complex (center). Unfortunately, subsequent thermodynamic analysis in the context of a test tube reveals that the desired heterodimer occurs at negligible concentration relative to other undesired monomers and homodimers (right). (b) Test tube design. Sequence design formulated in the context of a test tube (left) ensures that at equilibrium the desired 'on-target' complex is dominated by its target structure and forms at approximately its target concentration, and that undesired 'off-target' complexes (all monomers and homodimers) form at negligible concentrations (center). Subsequent thermodynamic analysis in the context of a test tube (right) is consistent with the test tube design formulation, hence providing no new information and no unpleasant surprises.

tube is then:

$$\Psi = \Psi_{\text{on}} \cup \Psi_{\text{off}}.$$

Each complex, $j \in \Psi$, is specified as a strand ordering, π_j, corresponding to structural ensemble $\Gamma(\pi_j)$. For each on-target complex, $j \in \Psi_{\text{on}}$, the user specifies a target secondary structure, s_j, and a target concentration, y_j. For each off-target complex, $j \in \Psi_{\text{off}}$, the target concentration is vanishing ($y_j = 0$) and there is no target structure ($s_j = \emptyset$). When specifying the off-targets in Ψ_{off}, it is convenient to include all complexes of up to $L_{\max}$ strands. For example, by (2.3), four strands can interact to form 108 complexes of up to size four.

Complementarity constraints may be imposed on the design at the sequence level by defining strands in terms of *sequence domains* and at the structural level by specifying base pairing within the on-target structures. Complementarity constraints can propagate between complexes if, for example, nucleotides a are b are paired in one on-target structure and nucleotides b and c are paired in another on-target structure.

3.1.2 Test tube ensemble defect objective function

We seek to perform sequence optimization for test tube design based on a physically meaningful objective function that quantifies sequence quality with respect to the target test tube.

As a precedent for this approach, consider the related problem of *complex design*, where the goal is to design strands that, at equilibrium, adopt a target secondary structure within the ensemble of a complex. For a candidate sequence, ϕ_j, and target structure, s_j, the *complex ensemble defect* [15, 83]

$$n(\phi_j, s_j) = |\phi_j| - \sum_{\substack{1 \le a \le |\phi_j| \\ 1 \le b \le |\phi_j|+1}} P^{a,b}(\phi_j) S(s_j), \tag{3.1}$$

is the average number of incorrectly paired nucleotides at equilibrium evaluated over the ensemble of the complex, Γ_j. The complex ensemble defect falls in the interval $(0, |\phi_j|)$. For complex design, the complex ensemble defect provides a physically meaningful objective function for quantifying sequence quality.

Here, to provide a basis for test tube design, we derive the test tube ensemble defect, representing the concentration of incorrectly paired nucleotides at equilibrium evaluated over the ensemble of the test tube. For a target test tube with target secondary structures, s_Ψ, target concentrations, y_Ψ, and candidate sequences, ϕ_Ψ, the *test tube ensemble defect*

$$C(\phi_\Psi, s_\Psi, y_\Psi) = \sum_{j \in \Psi} c(\phi_j, s_j, y_j) \tag{3.2}$$

may be expressed in terms of the defect contribution of each complex $j \in \Psi$:

$$c(\phi_j, s_j, y_j) = n(\phi_j, s_j) \min(x_j, y_j) + |\phi_j| \max(y_j - x_j, 0). \tag{3.3}$$

For each on-target complex $j \in \Psi_{\text{on}}$, the first term in (3.3) represents the *structural defect*, quantifying the concentration of nucleotides that are in an incorrect base pairing state on average within the ensemble of complex j, and the second term represents the *concentration defect*, quantifying the concentration of nucleotides that are in an incorrect base pairing state because there is a deficiency in the concentration of complex j. Because $y_j = 0$ for off-target complexes, the structural and concentration defects are both identically zero (so the sum in (3.2) may be written over Ψ_{on} instead of Ψ). This does not mean that the defects associated with the off-targets are ignored. By conservation of mass, non-zero off-target concentrations imply deficiencies in on-target concentrations, and these concentration defects are quantified by (3.3). The test tube ensemble defect falls in the interval

$(0, y_{\text{nt}})$, where

$$y_{\text{nt}} \equiv \sum_{j \in \Psi_{\text{on}}} |\phi_j| y_j$$

is the total concentration of nucleotides in the test tube.

Note that if there is only one species of complex in the test tube ($|\Psi| = 1$), its concentration is necessarily equal to the target concentration ($x_1 = y_1$), so the formulation is independent of concentration. In this case, optimization of the test tube ensemble defect, $C(\phi_1, s_1, y_1)$, is equivalent to optimization of the complex ensemble defect, $n(\phi_1, s_1)$.

Calculation of the test tube ensemble defect (3.2) requires calculation of the complex partition functions, Q_Ψ, which are used to calculate the equilibrium concentrations, x_Ψ, as well as the equilibrium pair probability matrices, $P_{\Psi_{\text{on}}}$, which are used to calculate the complex ensemble defects, $n_{\Psi_{\text{on}}}$. Hence, the time complexity to evaluate the test tube ensemble defect is the same as the time complexity to analyze equilibrium base pairing in a test tube.

3.2 Algorithm

3.2.1 Overview

We describe a test tube design algorithm based on test tube ensemble defect optimization. For a target test tube with target secondary structures, s_Ψ, and target concentrations, y_Ψ, we seek to design a set of sequences, ϕ_Ψ, such that the test tube ensemble defect satisfies the *test tube stop condition*:

$$C(\phi_\Psi, s_\Psi, y_\Psi) \leq C_{\text{stop}} \tag{3.4}$$

with

$$C_{\text{stop}} \equiv f_{\text{stop}} y_{\text{nt}} \tag{3.5}$$

for a user-specified value of $f_{\text{stop}} \in (0, 1)$.

The test tube ensemble defect is reduced via iterative mutation of a random initial sequence. Because of the high computational cost of calculating the test tube ensemble defect, it is important to avoid direct recalculation of C in evaluating each candidate mutation. We exploit two approximations to enable efficient estimation of the test tube ensemble defect: using *test tube ensemble focusing*, sequence optimization initially focuses on only the on-target portion of the test tube ensemble; using *hierarchical ensemble decomposition*, the structural ensemble of each on-target complex is hierarchically decomposed into a tree of subensembles, yielding a forest of decomposition trees. Candidate sequences are evaluated at the leaf level of the decomposition forest by estimating the test tube ensemble defect from nodal properties calculated efficiently over the leaf subensembles. As optimized subsequences are merged toward the root level of the forest, decomposition defects that

arise due to crosstalk between subsequences are eliminated via ensemble redecomposition from the parent level on down and sequence reoptimization from the leaf level on up. After subsequences are successfully merged to the root level, the full test tube ensemble defect, C, is calculated for the first time, including all on- and off-target complexes in the test tube ensemble. Any off-target complexes that form at appreciable concentration are hierarchically decomposed, added to the decomposition forest, and actively destabilized during subsequent forest reoptimization. The elements of this hierarchical sequence design algorithm are described below and detailed in the pseudocode of Algorithm 3.1.

3.2.2 Test tube ensemble focusing

To reduce the cost of sequence optimization, the set of complexes, Ψ, is partitioned into two disjoint sets:

$$\Psi = \Psi_{\text{active}} \cup \Psi_{\text{passive}} \tag{3.6}$$

where Ψ_{active} denotes complexes that will be actively designed, and Ψ_{passive} denotes complexes that will inherit sequence information from Ψ_{active}. Initially, we set

$$\Psi_{\text{active}} = \Psi_{\text{on}}, \quad \Psi_{\text{passive}} = \Psi_{\text{off}},$$

so that only the on-target complexes are included in the focused test tube ensemble at the outset of sequence design.

3.2.3 Hierarchical decomposition of on-target structures

Exact evaluation of the test tube ensemble, C, requires calculation of the defect contribution, c_j, for each complex $j \in \Psi_{\text{active}}$. The $\Theta(|\phi_j|^3)$ cost of calculating c_j is dominated by calculation of the partition function, Q_j, and equilibrium pair probability matrix, P_j. To reduce the cost of evaluating candidate sequences, we seek to estimate c_j at lower cost by hierarchically decomposing the structural ensemble Γ_j of each complex $j \in \Psi_{\text{active}}$ into a tree of subensembles, yielding a forest of $|\Psi_{\text{on}}|$ decomposition trees. Estimating the defect contribution, c_j, using physical quantities calculated at depth d in the decomposition forest requires calculation of the nodal partition function, $\tilde{Q}_k$, and nodal pair probability matrix, $\tilde{P}_k$, at cost $\Theta(|\phi_k|^3)$ for each node k at depth d. For these calculations, a bonus free energy is applied to base pairs sandwiching the split point to make them form with high probability, thus enforcing the decomposition assumption. For an optimal binary decomposition, $|\phi_k|$ halves and the number of nodes doubles at each depth moving down the tree, so the cost of estimating c_j at depth d can be a factor of $1/2^{2(d-1)}$ lower than the cost of calculating c_j exactly on the full ensemble Γ_j. Hence, for maximal efficiency, candidate mutationts are evaluated

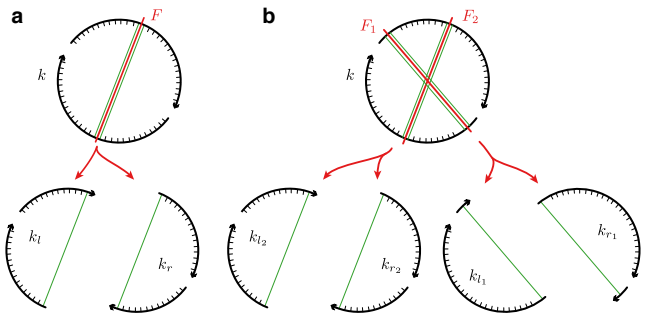


Figure 3.2: Ensemble decomposition of a parent node using one or more split-points sandwiched between base pairs. (a) A single split-point F partitions the nucleotides of parent node k into child nodes k_l and k_r. (b) The two split-points, F_1 and F_2, cross in the polymer graph and are hence exclusive (denoted $F_1 \otimes F_2$). Split-points within each parent are depicted in red, nucleotides sandwiching the split point are depicted in green in the children, and the remaining nucleotides are depicted in black.

based on the estimated test tube ensemble defect calculated at the leaves of the decomposition forest. As designed subsequences are merged toward the root level, the test tube ensemble defect is estimated at intermediate depths in the forest.

To decompose the structural ensemble, Γ_k, of parent node k, the nucleotides of k are partitioned into left and right child nodes, k_l and k_r, by a *split-point*, F (Figure 3.2a). The resulting child ensembles, Γ_{k_l} and Γ_{k_r}, can be used to reconstruct a strict subset, Γ_k^F, of the structures in the parent ensemble, Γ_k, (a subset that excludes all structures with base pairs crossing F in the polymer graph). For the purposes of accuracy, it is important that Γ_k^F should include those structures that dominate the equilibrium physical properties of Γ_k, while for the purposes of efficiency, it is important that Γ_k^F should exclude as many structures as possible that contribute negligibly to the equilibrium physical properties of Γ_k. Hence, the utility of ensemble decomposition hinges on suitable placement of the split-point F within parent node k.

The dual goals of accuracy and efficiency can both be achieved by placing the split point, F, within a duplex that forms with high probability at equilibrium such that approximately half the parent nucleotides are partitioned to each child. Recall that the structural ensemble, Γ_k, is defined to contain all unpseudoknotted secondary structures, corresponding to precisely those polymer graphs with no crossing base pairs. Since no structure can contain both base pairs that sandwich F and base pairs that cross F, placement of F between base pairs with probability close to one implies that the structures containing base pairs crossing F occur with low probability at equilibrium and may be safely neglected; it is exactly these structures with base pairs crossing F that are excluded from Γ_k^F. Partitioning the parent nucleotides into left and right children of equal size minimizes the total cost, $\Theta(|\phi_{k_l}|^3) + \Theta(|\phi_{k_r}|^3)$, of evaluating both children.

During the course of sequence design, if the base pairs sandwiching the split-point F in parent

k do not form with probability close to one, the accuracy of the decomposition breaks down. In this case, Γ_k^F excludes structures that are important to the equilibrium physical properties of Γ_k, preventing the children from approximating the full defect of the parent. As we describe later, decomposition defects are overcome by redecomposing the parental ensemble, Γ_k, using a set of exclusive split-points, $\{F\}$, that define exclusive child subensembles (Figure 3.2b), again enabling accurate estimation of the parent physical properties.

3.2.3.1 Structure-guided decomposition of on-target complexes

At the outset of sequence design, equilibrium base pairing probabilities are not yet available to guide ensemble decomposition. Instead, initial decomposition of each on-target complex, $j \in \Psi_{\text{active}}$, is guided by the user-specified on-target structure, s_j, making the optimistic assumption that the base pairs in s_j will form with probability close to one following sequence design. Using this structure-guided ensemble decomposition approach, as the quality of the sequence design improves, the quality of the ensemble decomposition approximation will also improve.

Each target structure, $s_j \in \Psi_{\text{active}}$, is decomposed into a (possibly unbalanced) binary tree of substructures, resulting in a forest of $|\Psi_{\text{on}}|$ trees. Each node in the forest is indexed by a unique integer k. For each parent node, k, there is a left child node, k_l, and a right child node, k_r. Each nucleotide in parent structure s_k is partitioned to either the left or right child substructure ($s_k = s_{k_l} \cup s_{k_r}, s_{k_l} \cap s_{k_r} = \emptyset$) via decomposition at a split-point F between base pairs within a duplex stem of s_k.

Eligible *split-points* are those locations within a duplex stem with at least H_{split} consecutive base pairs on either side, such that each child would have at least N_{split} nucleotides. An eligible split-point is selected so as to minimize the difference in the number of nucleotides in each child, $||\phi_{k_l}| - |\phi_{k_r}||$. See Figure 3.3 for an example of structure-guided decomposition. Decomposition of the sequence, ϕ_k, is performed in accordance with decomposition of structure s_k.

If the maximum depth of a leaf in the forest of binary trees is D, any nodes with depth $d < D$ that lack an eligible split-point are replicated at each depth down to D so that all leaves have depth D. Let Λ denote the set of all nodes in the forest. Let Λ_d denote the set of all nodes at depth d. Let $\Lambda_{d,j}$ denote the set of all nodes at depth d resulting from decomposition of complex j. Each nucleotide in complex j is present in exactly one nodal sequence, $\phi_k \in \phi_{\Lambda_{d,j}}$, at any depth $d \in 1, \ldots, D$.

3.2.3.2 Stop condition stringency

In order to build in tolerance for a basal level of decomposition defect as subsequences are merged moving up the decomposition forest, the stringency of the test tube stop condition (3.5) is increased

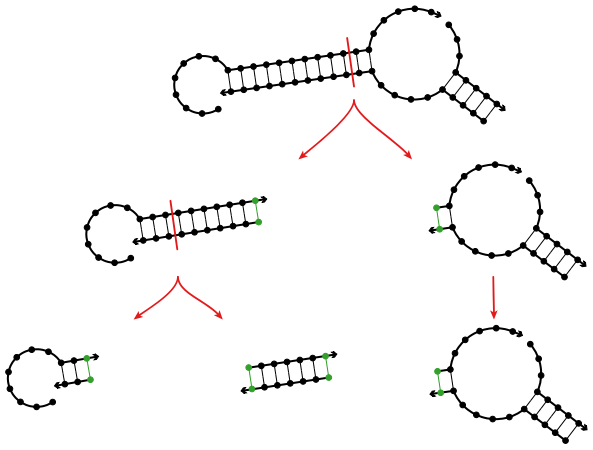


Figure 3.3: Hierarchical decomposition of an on-target structure. The selected split-point within each parent is denoted by a red line. The base pairs sandwiching split points are depicted in green in the children. The remaining nucleotides within each structure are depicted in black. $H_{\text{split}} = 2, N_{\text{split}} = 12$.

by a factor of $f_{\text{stringent}} \in (0, 1)$ at each level $d \in \{2, \ldots, D\}$ moving down the decomposition forest

$$C_d^{\text{stop}} \equiv C_{\text{stop}}(f_{\text{stringent}})^{d-1}. \tag{3.7}$$

3.2.4 Test tube ensemble defect estimation from nodal contributions

In the following sections, we describe how to calculate each of the nodal contributions at any level $d \in \{2, \ldots, D\}$ so as to efficiently and accurately estimate the complex contributions, $c_{\Psi_{\text{active}}}$, to the test tube ensemble defect. We describe how to construct the complex partition function estimates, $\tilde{Q}_{\Psi_{\text{active}}}$, and pair probability matrices, $\tilde{P}_{\Psi_{\text{active}}}$, using nodal partition functions, Q_{Λ_d}, and probability matrices, P_{Λ_d}. Complex concentration estimates, $x_{\tilde{\Psi}_{\text{active}}}$, are then calculated based on $\tilde{Q}_{\Psi_{\text{active}}}$, using deflated mass constraints to model the effect of the neglected off-target complexes in Ψ_{passive}. Complex ensemble defect estimates, $\tilde{n}_{\Psi_{\text{on}}}$, are calculated based on $\tilde{P}_{\Psi_{\text{active}}}$. The concentration and complex ensemble defect estimates are then used to calculate the complex contributions to the test tube ensemble defect, $\tilde{c}_{\Psi_{\text{active}}}$, which are summed to produce the test tube ensemble defect estimate, $\tilde{C}_d$.

3.2.4.1 Complex partition function estimate

We begin by calculating the complex partition function estimate, $\tilde{Q}_j$, for each complex $j \in \Psi_{\text{active}}$ in terms of partition function contributions evaluated efficiently at the nodes $k \in \Lambda_{d,j}$ at any level $d \in \{2, \ldots, D\}$. This decomposition is illustrated for parent node k and its children k_l and k_r in Figure 3.4a.

Let E_k denote the set of base pairs adjacent to split points in node k. For each base pair

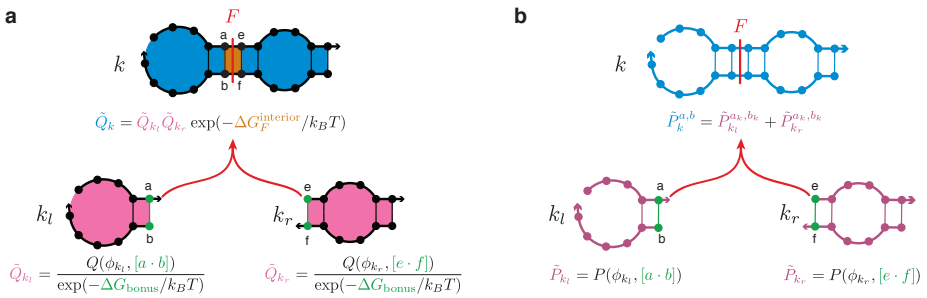


Figure 3.4: Efficient estimation of physical quantities from nodal contributions. (a) Complex partition function estimate. Conceptually, $\tilde{Q}_{k_l}$, the partition function estimate of child k_l (calculated on child node k_l at cost $\Theta(|\phi_{k_l}|^3)$, approximates the contribution of the left-child nucleotides to the partition function of parent k (which can be calculated exactly on parent node k at higher cost $\Theta(|\phi_k|^3)$). (b) Pair probability estimates. Conceptually, $\tilde{P}_{k_l}$, the pair probabilities of child k_l (calculated on child node k_l at cost $\Theta(|\phi_{k_l}|^3)$, approximates the contribution of the left-child nucleotides to pair probabilities of parent k (which can be calculated exactly on parent node k at higher cost $\Theta(|\phi_k|^3)$).

$[a \cdot b] \in E_k$, we apply a bonus free energy, ΔG_{bonus}, to all structures containing the base pair while calculating the nodal partition function, $Q(\phi_k, E_k)$. The nodal partition function estimate $\tilde{Q}_k$ is then calculated by dividing bonus contributions from the nodal partition function:

$$\tilde{Q}_k = Q(\phi_k, E_k)/\exp(-|E_k|\Delta G^{\text{bonus}}/k_BT). \tag{3.8}$$

As ΔG^{bonus} is made more favorable, the probabilities of base pairs in E_k approach unity and the nodal partition function estimate approaches the corresponding partition function over only structures containing all pairs in E_k. The partition functions, $Q(\phi_k)$, are calculated using dynamic programs suitable for complexes containing arbitrary numbers of strands [14].[1]

Next, the partition functions calculated at depth d are merged recursively to estimate the partition function for complex j. Consider split-point F in parent k, with left-child and right-child partition functions $\tilde{Q}_{k_l}$ and $\tilde{Q}_{k_r}$, and free energy, $\Delta G_F^{\text{interior}}$, for the interior loop formed by the base

[1]The periodic strand repeat, v_j, of complex j is defined as the number of different rotations of the polymer graph that map strands of the same type to each other (e.g., $v_j = 4$ for complex AAAA, $v_j = 3$ for complex ABABAB, $v_j = 2$ for ABAABA). For complexes in which all strands are distinct, $v_j = 1$. However, complexes containing multiple copies of the same strand may have $v_j > 1$, in which case the dynamic program that is used to calculate the partition function of complex j will be incorrect due to symmetry and overcounting errors that are different for different structures in Γ_j. Fortunately, these errors interact in such a way that they can be exactly and simultaneously corrected by dividing the calculated partition function by the integer v_j

$$Q(\phi_j) = Q_{\text{calc}}(\phi_j)/v_j.$$

When employing the dynamic program to calculate the nodal partition functions for $k \in \Lambda_{d,j}$, it is important to correct each of these values using v_j.

pairs sandwiching F. The partition function estimate for parent k is then[2]

$$\tilde{Q}_k = \tilde{Q}_{k_l}\tilde{Q}_{k_r}\exp(-\Delta G_F^{\text{interior}}/k_BT). \tag{3.9}$$

The expression becomes exact as the equilibrium probabilities of the base pairs sandwiching F approach unity, which is enforced for the children as ΔG^{bonus} is made more favorable. At the conclusion of recursive merging, the partition function estimate for complex j based on information calculated at depth d is then

$$\tilde{Q}_j = \tilde{Q}_k \tag{3.10}$$

where $\{k\} = \Lambda_{1,j}$ is the root node of the decomposition forest.

3.2.4.2 Complex concentration estimate using deflated mass constraints

After calculating the set of complex partition function estimates, $\tilde{Q}_{\Psi_{\text{active}}}$, based on the nodal partition function contributions at level d, the corresponding equilibrium complex concentration estimates, $\tilde{x}_{\Psi_{\text{active}}}$, may be found by solving the convex programming problem (2.2a). To impose the conservation of mass constraints (2.2b), the total concentration of each strand species, $i \in \Psi^0$, must be specified. The total strand concentrations,

$$x_i^0 = \sum_{j\in\Psi_{\text{on}}} A_{i,j}y_j \quad \forall i \in \Psi^0, \tag{3.11}$$

follow from the target concentrations, y_Ψ, and strand composition, $A_{i,j}$, of each on-target complex $j \in \Psi_{\text{on}}$.

Using test tube ensemble focusing, initial sequence optimization is performed on a decomposition forest that contains only the on-target complexes in Ψ_{active}, but ultimately, we wish to satisfy the test tube stop condition (3.4) for the full set of complexes in Ψ, including the off-targets in Ψ_{passive}. Recall that the off-targets in Ψ_{passive} do not contribute directly to the sum used to calculate the test tube ensemble defect (3.2), but contribute indirectly by forming at positive concentrations, causing concentration defects for complexes in Ψ^{active} as a result of conservation of mass. Hence, we can pre-allocate a portion of the permitted test tube ensemble defect, $f_{\text{stop}}y_{\text{nt}}$, to the neglected off-target complexes in Ψ_{passive} by deflating the total strand concentrations (3.11) used to impose the mass constraints in calculating the equilibrium concentrations $\tilde{x}_{\Psi_{\text{active}}}$.

Following this approach, if $\Psi_{\text{passive}} \neq \emptyset$, we make the assumption that the complexes in Ψ_{passive}

[2]During merging, free energies contributions inherent in the sandwiching base pairs terminating a helix must be corrected. In particular, there is a free energy penalty for ending a duplex in a non-[G·C] base pair in the current free energy models. This is corrected during the merge step since the base pairs no longer terminate duplex.

consume a constant fraction of each total strand concentration:

$$\sum_{j \in \Psi_{\text{passive}}} A_{i,j} \tilde{x}_j = f_{\text{passive}} f_{\text{stop}} \sum_{j \in \Psi_{\text{on}}} A_{i,j} y_j \quad \forall i \in \Psi^0,$$

corresponding to a total mass allocation of $f_{\text{passive}} f_{\text{stop}} y_{\text{nt}}$ to the neglected off-targets in Ψ_{passive}. To estimate the equilibrium concentrations of the complexes in Ψ_{active} via (2.2a), we therefore use the deflated strand concentrations:

$$x_i^0 = (1 - f_{\text{passive}} f_{\text{stop}}) \sum_{j \in \Psi_{\text{on}}} A_{i,j} y_j \quad \forall i \in \Psi^0 \tag{3.12}$$

in place of the full strand concentrations (3.11).

3.2.4.3 Complex ensemble defect estimate

For each complex $j \in \Psi_{\text{active}}$, the complex pair probability estimate, $\tilde{P}_j$, is calculated from the nodal pair probability matrices, $P(\phi_k)$, calculated efficiently at nodes $k \in \Lambda_{d,j}$ at any level $d \in \{2, \ldots, D\}$. The complex ensemble defect estimate $\tilde{n}_j$ is calculated using $\tilde{P}_j$. Estimation of pair probabilities is illustrated for parent node k and its children k_l and k_r in Figure 3.4b.

Because each nucleotide in complex j is present in exactly one node, $k \in \Lambda_{d,j}$, we can approximate the complex pair probabilities as a mapping of the nodal pair probability matrices at any depth in the subtree. The nodal pair probability matrix, $P(\phi_k)$, was previously calculated during nodal partition function estimation (3.8). During this calculation, a free energy bonus is applied to the formation of each base pair that sandwiches a split point $[a \cdot b] \in E_k$. As the free energy bonus is made more favorable, the pair probabilities converge to the pair probabilities calculated over only structures that contain all base pairs in E_k.

The nodal pair probabilities calculated at level d are merged recursively to estimate the pair probabilities for complex j. Consider parent k, with left-child and right-child pair probabilities $\tilde{P}_{k_l}$ and $\tilde{P}_{k_r}$. We define $\tilde{P}_{k_l}^{a_k, b_k}$ to be the pair probability of base pair $[a \cdot b]$ in parent node k according to child k_l. If both bases are present in the child, this corresponds to an entry in the child's pair probability matrix. Otherwise, the child cannot capture this contribution to the parental pair probabilities, so it is zero. Using this, the parental pair probabilities can be estimated by merging the child probabilities

$$\tilde{P}_k^{a,b} = \tilde{P}_{k_l}^{a_k, b_k} + \tilde{P}_{k_r}^{a_k, b_k}. \tag{3.13}$$

At the conclusion of recursive merging, the pair probability estimate for complex j based on the

information calculated at depth d is then

$$\tilde{P}_j = \tilde{P}_k \tag{3.14}$$

where $\tilde{P}_k$ is the pair probabilities of nodes $k \in \Lambda_{d,j}$, merged to the root node, $k : \{k\} = \Lambda_{1,j}$, of the decomposition forest.

The complex ensemble defect for complex j can then be estimated using the pair probability matrix estimate:

$$\tilde{n}_j = |\phi_j| - \sum_{\substack{1 \le a \le |\phi_j| \\ 1 \le b \le |\phi_j|+1}} \tilde{P}_j^{a,b} S(s_j). \tag{3.15}$$

This estimate becomes exact in the limit as the equilibrium probabilities of the base pairs sandwiching the decomposition split-points approach unity. The contribution of nucleotide a to the complex ensemble defect of complex j is given by

$$\tilde{n}_j^a = 1 - \sum_{1 \le b \le |\phi_j|+1} P_j^{a,b} S_j^{a,b}. \tag{3.16}$$

As the bonus free energy is made more favorable for nucleotides sandwiching the split points, the nucleotide defects at these locations approach zero, enforcing the decomposition assumption; base pairs in long helices form with high probability. If this assumption is correct, the low defect in these base pairs will be an accurate portrayal of parental properties. If this assumption is incorrect, a new set of split points will be chosen during redecomposition, as described later.

3.2.4.4 Test tube ensemble defect estimate

Having calculated the complex concentration estimates, $\tilde{x}_{\Psi_{\text{active}}}$, and the complex ensemble defect estimates, $\tilde{n}_{\Psi_{\text{active}}}$, based on nodal contributions at any depth $d \in \{2, \ldots, D\}$, the contribution of complex $j \in \Psi_{\text{active}}$ to the test tube ensemble defect is

$$\tilde{c}_j = \tilde{n}_j \min\left(\tilde{x}_j, y_j\right) + |\phi_j| \max\left(y_j - \tilde{x}_j, 0\right), \tag{3.17}$$

and the test tube ensemble defect estimate is:

$$\tilde{C}_d = \sum_{j \in \Psi_{\text{active}}} \tilde{c}_j. \tag{3.18}$$

This sum can equivalently be expressed as a sum over nucleotide contributions at depth d. The test tube ensemble defect associated with nucleotide a in complex $j \in \Psi_{\text{active}}$ is

$$\tilde{c}_j^a = \tilde{n}_j^a \min(\tilde{x}_j, y_j) + \max(y_j - \tilde{x}_j, 0)$$

and the test tube ensemble defect estimate (3.18) becomes:

$$\tilde{C}_d = \sum_{j \in \Psi_{\text{active}}} \sum_{a \in 1,\ldots,|\phi_j|} c_j^a. \tag{3.19}$$

3.2.5 Sequence optimization at the leaves of the decomposition forest

3.2.5.1 Initialization

The sequences for the complexes $j \in \Psi_{\text{active}}$ are randomly initialized subject to complementarity constraints in the design problem specification: Watson-Crick complements are used to initialize complementary sequence domains or any bases that are paired within an on-target structure. These initial sequences are pushed down to the leaf level of the decomposition forest.

3.2.5.2 Leaf mutation

To reduce computational cost, all candidate mutations are evaluated at the leaf nodes, $k \in \Lambda_D$, of the decomposition forest. Leaf mutation terminates successfully if the *leaf stop condition*,

$$\tilde{C} \leq C_D^{\text{stop}}, \tag{3.20}$$

is satisfied. A candidate mutation is accepted if it decreases the test tube ensemble defect estimate (3.18) and rejected otherwise.

We perform *defect weighted mutation sampling* by selecting nucleotide a for mutation with probability proportional to the contribution of nucleotide a to the defect at level d: $c_j^a/\tilde{C}_D$. If the selected candidate mutation position is subject to complementarity constraints implied by the design problem specification, either via complementary sequence domains or via base pairing within any on-target structure, the candidate mutation respects the constraint in either Watson–Crick complementarity (default; constrained nucleotides are selected randomly from a uniform distribution of Watson–Crick pairs) or wobble complementarity (constrained nucleotides are selected randomly from a uniform distribution of Watson–Crick and wobble pairs). For design problems where on-targets place competing demands on the test tube ensemble defect, permitting wobble complementarity gives the algorithm additional flexibility in meeting these demands (see Figure 3.11).

A candidate sequence $\hat{\phi}_{\Lambda_D}$ is evaluated via calculation of the test tube ensemble defect estimate, $\tilde{C}_D$, if the candidate mutation, ξ, is not in the set of previously rejected mutations, $\gamma_{\text{unfavorable}}$ (position and sequence). The set, $\gamma_{\text{unfavorable}}$, is updated after each unsuccessful mutation and cleared after each successful mutation. A counter, $m_{\text{unfavorable}}$, is used to keep track of the number of consecutive failed mutation attempts; it is incremented after each unsuccessful mutation and reset to zero after each successful mutation. Leaf mutation terminates unsuccessfully if $m_{\text{unfavorable}} \geq$

$M_{\text{unfavorable}}$. The outcome of leaf mutation is the set of leaf sequences, ϕ_{Λ_D}, corresponding to the lowest encountered $\tilde{C}_D$.

3.2.5.3 Leaf reoptimization

After leaf mutation terminates, if the leaf stop condition (3.20) is not satisfied, leaf reoptimization commences. Sequences are perturbed by using defect weighted mutation sampling to pick M_{reseed} distinct nucleotides to mutate. These locations, and any complementary locations, are reseeded and a new round of leaf mutation is performed, starting at the perturbed sequences. The reoptimized candidate sequences, $\hat{\phi}_{\Lambda_D}$, are accepted if they decrease $\tilde{C}_D$ and rejected otherwise. The counter m_{reopt} is used to keep track of the number of consecutive rejections; it is incremented after each rejection and reset to zero upon sequence acceptance. Leaf reoptimization terminates successfully if the leaf stop condition is satisfied, and unsuccessfully if $m_{\text{reopt}} \geq M_{\text{reopt}}$. The outcome of leaf reoptimization is the set of leaf sequences, ϕ_{Λ_D}, corresponding to the lowest encountered $\tilde{C}_D$.

3.2.6 Subsequence merging, redecomposition, and reoptimization

After leaf reoptimization terminates, parent nodes at depth $d = D - 1$ merge their left and right child sequences to create the set of candidate sequences $\hat{\phi}_{\Lambda_d}$. The parental test tube ensemble defect estimate, $\tilde{C}_d$ is calculated and the candidate sequences, $\hat{\phi}_{\Lambda_d}$, are accepted if they decrease $\tilde{C}_d$ and rejected otherwise. If the *parental stop condition*,

$$\tilde{C}_d \leq \max(C_d^{\text{stop}}, \tilde{C}_{d+1}/f_{\text{stringent}}), \tag{3.21}$$

is satisfied with $C_d^{\text{stop}} \equiv C_{\text{stop}}(f_{\text{stringent}})^{d-1}$, merging continues up to the next level in the forest. Otherwise, failure to satisfy the parental stop condition indicates the existence of a *decomposition defect*,

$$\tilde{C}_d - \tilde{C}_{d+1}/f_{\text{stringent}} > 0, \tag{3.22}$$

exceeding the basal level permitted by the parameter $f_{\text{stringent}}$. The parent node at depth d that results in the smallest decomposition defect when replaced by its children from depth $d + 1$ is subject to (structure- and probability-guided) hierarchical ensemble redecomposition, as described later. Additional parents are redecomposed until

$$\tilde{C}_d - \tilde{C}^*_{d+1}/f_{\text{stringent}} \leq f_{\text{redecomp}}(\tilde{C}_d - \tilde{C}_{d+1}/f_{\text{stringent}}), \tag{3.23}$$

where $\tilde{C}_{d+1}$ is the child defect estimate before any redecomposition, and $\tilde{C}^*_{d+1}$ is the child defect after redecomposition and $f_{\text{redecomp}} \in (0, 1)$.

After redecomposition, the current sequences at depth d are pushed down to all nodes below

depth d, and a new round of leaf mutation and leaf reoptimization is performed. Following leaf reoptimization, merging begins again. Subsequence merging and reoptimization terminate successfully when the parental stop condition (3.21) is satisfied at depth $d = 1$. The outcome of subsequence merging, redecomposition, and reoptimization is the set of sequences, $\phi_{\Psi_{\text{active}}}$, corresponding to the lowest encountered $\tilde{C}_1$.

3.2.7 Test tube evaluation, refocusing, and reoptimization

Using test tube ensemble focusing, initial sequence optimization is performed for the on-target complexes in Ψ_{active}, neglecting the off-target complexes in Ψ_{passive}. At the termination of initial forest optimization, the estimated test tube ensemble defect, $\tilde{C}_1$, is calculated using (3.18). For this estimate, the complex defect contributions, $\tilde{c}_{\Psi_{\text{active}}}$, are based on complex concentration estimates, $\tilde{x}_{\Psi_{\text{active}}}$, calculated using deflated total strand concentrations (3.12) to create a built-in defect allowance for the effect of the neglected off-target in Ψ_{passive}. The exact test tube ensemble defect, C, is then evaluated for the first time over the full ensemble Ψ using (3.2). For this exact calculation, the complex defect contributions, c_{Ψ}, are based on complex concentrations, x_{Ψ}, calculated using the full strand concentrations (3.11).

Sequence design terminates successfully if the test tube ensemble defect satisfies the termination stop condition:

$$C \leq \max\bigl(C_{\text{stop}}, \tilde{C}_1\bigr). \tag{3.24}$$

Otherwise, failure to satisfy the termination stop condition indicates the existence of a *focusing defect*:

$$C - \tilde{C}_1 > 0. \tag{3.25}$$

The test tube ensemble is then refocused by transferring the highest-concentration off-target in Ψ_{passive} to Ψ_{active}. Additional off-targets are transferred from Ψ_{passive} to Ψ_{active} until

$$\tilde{C} - \tilde{C}_1^* \leq f_{\text{refocus}}(C - \tilde{C}_1), \tag{3.26}$$

where $\tilde{C}_1$ is the forest-estimated defect before any refocusing and $\tilde{C}_1^*$ is the forest-estimated defect after refocusing (calculated using deflated strand concentrations if $\Psi_{\text{passive}} \neq \emptyset$, and $f_{\text{refocus}} \in (0, 1)$.

The new off-target structures in Ψ_{active} are then decomposed using probability-guided decomposition, as described later. The decomposition forest is augmented with new nodes at all depths, and forest reoptimization commences starting from the final sequences from the previous round of forest optimization. During forest reoptimization, the algorithm actively attempts to destabilize the off-targets that were added to Ψ_{active}, since the estimated concentrations of on-targets will decrease when off-target concentrations increase. This process of ensemble refocusing and forest reoptimiza-

tion is repeated until the termination stop condition (3.24) is satisfied, which is guaranteed to occur in the event that all off-targets are eventually added to Ψ_{active}. At the conclusion of sequence design, the algorithm returns the sequences, ϕ_Ψ, that yielded the lowest encountered test tube ensemble defect, C.

3.2.8 Hierarchical ensemble decomposition using multiple exclusive split-points

Prior to sequence design, in the absence of base pairing probability information, hierarchical ensemble decomposition was performed for each complex $j \in \Psi_{\text{active}}$ based on the user-specified on-target structure, s_j. For a parent node, k, with structural ensemble, Γ_k, a single split-point, F, was positioned within a duplex in target structure s_j, so as to minimize the cost of evaluating both children, yielding left and right child nodes k_l and k_r with ensembles Γ_{k_l}, and Γ_{k_r}. These child ensembles enable reconstructurion of a strict subset, Γ_k^F, of the structures in the parent ensemble, Γ_k, (a subset which excludes all structures with base pairs crossing F in the polymer graph). Following leaf optimization, when left and right child sequences are merged to form a parent sequence, if decomposition defects are observed, this is symptomatic of an inadequate ensemble decomposition; when the base pairs sandwiching F form in the parent with a probability below unity, then Γ_k^F excludes structures that are important to the equilibrium physical properties of Γ_k, preventing the child nodes from predicting the full defect contribution of the parent node. This situation is remedied by redecomposing the parent, taking into consideration the newly-available parental base pairing probabilities.

Two candidate split-points, F_i and F_j, are *exclusive* if they cross when depicted in a polymer graph (denoted $F_i \otimes F_j$; see Fig. 3.2b). The parental ensembles, $\Gamma_k^{F_i}$ and $\Gamma_k^{F_j}$, reconstructed from the child ensembles implied by exclusive split points, F_i and F_j, have no structures in common ($\Gamma_k^{F_i} \cap \Gamma_k^{F_j} = \emptyset$). Hence, if a single split-point is inadequate for ensemble decomposition of a parent (i.e., the sandwiching base pairs form with probability substantially below one), a set of mutually exclusive split-points, $\{F\}$, can be used to non-redundantly decompose the parent ensemble so that collectively, the probability of the base pairs sandwiching the split-points approaches unity from below. During subsequence merging, redecomposition of parent nodes derived from on-target complexes is performed using structure- and probability-guided decomposition with multiple exclusive split-points. During off-target destabilization, decomposition of parent nodes derived from off-target complexes (for which no target structures exist), is performed using probability-guided decomposition with multiple exclusive split points. In either case, selection of the optimal set of exclusive split points is determined using a branch and bound algorithm to minimize the cost of evaluating the implied child nodes (see section B.2). Because exclusive split-points lead to exclusive structural

ensembles, it is straightforward to generalize the expressions used to reconstruct parental physical properties, as detailed below.

3.2.8.1 Structure-guided decomposition using a single split-point

For comparison with the formulations that follow, here we recast structure-guided decomposition using modified notation. Let F denote a split point and let $F^{\pm}$ denote the union of the sets of H_{split} base pairs sandwiching F on either side. For a node, k, descendant from on-target complex $j \in \Psi_{\text{active}}$ with user-specified target structure s_j, the nodal target structure matrix, S_k, is defined using the corresponding entries from the root target structure matrix, S_j. The set of valid split-points may be denoted

$$B(S_k) \equiv \left\{ F : \begin{array}{c} \min\limits_{a \cdot b \in F^{\pm}} S_k^{a,b} = 1 \\ \\ \min(|\phi_{k_l}|, |\phi_{k_r}|) \geq N_{\text{split}} \end{array} \right\} \tag{3.27}$$

and the optimal split-point selected for decomposition,

$$F^* \equiv \min_{F \in B(S_k)} \left(|\phi_{k_l}|^3 + |\phi_{k_r}|^3 \right), \tag{3.28}$$

minimizes the cost of evaluating the two child nodes implied by F.

3.2.8.2 Probability-guided decomposition using multiple exclusive split points

The set of sets of valid exclusive splits points may be denoted

$$\bar{B}(P_k) \equiv \left\{ \{F\} : \begin{array}{c} f_{\text{split}} \leq \sum\limits_{F_i \in \{F\}} \min\limits_{a \cdot b \in F_i^{\pm}} P_k^{a,b} \\ \\ \min\limits_{F_i \in \{F\}} (|\phi_{k_{l_i}}|, |\phi_{k_{r_i}}|) \geq N_{\text{split}} \\ \\ F_i \otimes F_j \;\; \forall F_i \neq F_j \in \{F\} \end{array} \right\} \tag{3.29}$$

and the optimal set of exclusive split points selected for decomposition,

$$\{F\}^* \equiv \min_{\{F\} \in \bar{B}(P_k)} \sum_{F_i \in \{F\}} \left(|\phi_{k_{l_i}}|^3 + |\phi_{k_{r_i}}|^3 \right), \tag{3.30}$$

minimizes the cost of evaluating the $2|\{F\}|$ child nodes implied by $\{F\}$.

3.2.8.3 Structure- and probability-guided decomposition using multiple exclusive split points

The set of sets of valid split points may be denoted[3]

$$\hat{B}(S_k, P_k) \equiv \tag{3.31}$$

$$\left\{ \{F\} : \begin{array}{c} \{F\} = G_i \cup \{G\}_j, G_i \in B(S_k), \{G\}_j \in \bar{B}(P_k) \\ F_i \otimes F_j \ \ \forall F_i \neq F_j \in \{F\} \end{array} \right\} \tag{3.32}$$

and the optimal set of exclusive split-points selected for decomposition,

$$\{F\}^* \equiv \min_{\{F\} \in \hat{B}(S_k, P_k)} \sum_{F_i \in \{F\}} \left(|\phi_{k_{l_i}}|^3 + |\phi_{k_{r_i}}|^3 \right), \tag{3.33}$$

minimizes the cost of evaluating the $2|F|$ child nodes implied by $\{F\}$. The structure-guided component of this approach ensures that the redecomposition is compatible with the user-specified target structure, while the probability-guided component of this approach ensures that the physical properties of the parent can be accurately estimated using the children.

3.2.9 Test tube ensemble defect estimation using multiple exclusive split points

Here, we generalize the formulation of test tube ensemble defect estimation at depth $d \in \{2, \ldots, D\}$ to account for the possibility of multiple exclusive split-points within any parent in the decomposition forest. Consider parent k decomposed using the set of exclusive split-points $\{F\}$. Following (3.9), for each split-point, $F_i \in \{F\}$, the corresponding exclusive contribution to the parental partition function is

$$\tilde{Q}_{k_i} = \tilde{Q}_{k_{l_i}} \tilde{Q}_{k_{r_i}} \exp(-\Delta G_{F_i}^{\text{interior}} / k_B T), \tag{3.34}$$

yielding the partition function estimate for parent k:

$$\tilde{Q}_k = \sum_{F_i \in \{F\}} \tilde{Q}_{k_i}. \tag{3.35}$$

Recursive merging is performed until the complex partition function estimate is obtained using (3.10).

At each recursive step, we simultaneously merge the nodal pair probabilities. For each split-point,

[3]Note that for a child node, k, generated using multiple exclusive split-points, the target structure matrix, S_k, may have row sums not equal to unity; if a nucleotide a in node k is intended to form an on-target base pair with a nucleotide not contained in node k, this will cause S_k to be zero for all entries in row a.

Parameter	RNA	
H_{split}	2	
N_{split}	12	
f_{split}	0.995	
$f_{\text{stringent}}$	0.995	
f_{redecomp}	0.9	
f_{refocus}	0.9	
f_{passive}	0.05	
M_{leaf}	6	
$M_{\text{unfavorable}}$	200	
M_{reseed}	50	
ΔG_{bonus}	-25	kcal/mol

Table 3.1: Default parameters for RNA sequence design. For DNA sequence design, $H_{\text{split}} = 3$.

$F_i \in \{F\}$, the corresponding exclusive pair probability contributions are

$$\tilde{P}_{k_i}^{a,b} = \tilde{P}_{k_{l_i}}^{a_k,b_k} + \tilde{P}_{k_{r_i}}^{a_k,b_k}. \tag{3.36}$$

Weighting each contribution by its exclusive contribution to the parental partition function yields pair probability matrix estimate for parent k:

$$\tilde{P}_{k}^{a,b} = \sum_{F_i \in \{F\}} \tilde{P}_{k_i}^{a,b} \frac{\tilde{Q}_{k_i}}{\tilde{Q}_k}. \tag{3.37}$$

Recursive merging is performed until the complex pair probability estimates are obtained using (3.14). Together, the complex partition function estimate and pair probability estimates enable calculation of complex concentration estimates (Section 3.2.4.2) and complex ensemble defect estimates (3.15), from which the test tube defect estimate at level d can be calculated (3.18).

3.3 Methods

3.3.1 Implementation

The test tube design algorithm is coded in the C programming language. The algorithm is available for non-commercial research purposes via the NUPACK web application and code base (`www.nupack.org`) [82].

3.3.2 Target test tubes

Algorithm performance is demonstrated using a set of target test tubes. For the *engineered test set*, each on-target structure was randomly generated with stem and loop sizes randomly seelcted from a distribution of sizes representative of the nucleic acid engineering literature. For the *random test*

OptimizeTube($s_\Psi, y_\Psi, \Psi, \Psi_{\text{on}}, \Psi_{\text{off}}$)
  $\phi_\Psi \leftarrow$ InitSeq($\emptyset, s_\Psi, \Psi$)
  $\Psi_{\text{active}}, \Psi_{\text{passive}} \leftarrow \Psi_{\text{on}}, \Psi_{\text{off}}$
  $\phi_\Lambda, \Lambda, D \leftarrow$ Decompose($\phi_{\Psi_{\text{active}}}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  $\phi_\Psi, \tilde{C}_1 \leftarrow$ OptimizeForest($\phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $C \leftarrow$ EvaluateDefect($\phi_\Psi, s_\Psi, y_\Psi$)
  $\hat{\phi}_\Psi, \hat{C} \leftarrow \phi_\Psi, C$
  while $\hat{C} > \max(C_{\text{stop}}, \tilde{C}_1)$
    $s_{\Psi_{\text{active}}} \leftarrow$ AugmentActive($s_{\Psi_{\text{active}}}, \hat{C}, \tilde{C}_1, \hat{\phi}_\Psi$)
    $\phi_\Lambda, \Lambda, D \leftarrow$ Decompose($\phi_{\Psi_{\text{active}}}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
    $\hat{\phi}_\Psi, \tilde{C}_1 \leftarrow$ OptimizeForest($\phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
    $\hat{C} \leftarrow$ EvaluateDefect($\hat{\phi}_\Psi, s_\Psi, y_\Psi$)
    if $\hat{C} < C$
      $C, \phi_\Psi \leftarrow \hat{C}, \hat{\phi}_\Psi$
  return ϕ_Ψ

OptimizeForest($\phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\tilde{C}_{1,\ldots,D} \leftarrow \infty$
  $C_d^{\text{stop}} \leftarrow C_{\text{stop}}(f_{\text{stringent}})^{d-1}\ \forall d \in \{1, \ldots, D\}$
  pstop $\leftarrow$ **false**
  while ¬pstop
    $\phi_{\Lambda_D}, \tilde{C}_D \leftarrow$ OptimizeLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
    $d \leftarrow D - 1$
    pstop $\leftarrow$ **true**
    while $d \geq 1$ **and** pstop
      $\hat{\phi}_{\Lambda_d} \leftarrow$ MergeSeq($\phi_{\Lambda_{d+1}}$)
      $\hat{C}_d, \hat{c}_{\Psi_{\text{active}}} \leftarrow$ EstimateDefect($\hat{\phi}_{\Lambda_d}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
      if $\hat{C}_d < \tilde{C}_d$
        $\phi_{\Lambda_d}, \tilde{C}_d \leftarrow \hat{\phi}_{\Lambda_d}, \hat{C}_d$
      if $\hat{C}_d > \max(C_d^{\text{stop}}, \tilde{C}_{d+1}/f_{\text{stringent}})$
        pstop $\leftarrow$**false**
        $\Lambda, \hat{\phi}_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}} \leftarrow$ Redecompose($\Lambda, \hat{\phi}_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
        for $d' = d+1, \ldots, D$
          $\phi_{\Lambda_{d'}} \leftarrow$ SplitSeq($\hat{\phi}_{\Lambda_{d'-1}}$)
          $\tilde{C}_{d'} \leftarrow \infty$
      $d \leftarrow d - 1$
  return $\phi_{\Lambda_1}, \tilde{C}_1$

AugmentActive($s_{\Psi_{\text{active}}}, C, \tilde{C}_1, \phi_\Psi$)
  $\tilde{C}_1^* \leftarrow \tilde{C}_1$
  while $C - \tilde{C}_1^* > f_{\text{refocus}}(C - \tilde{C}_1)$
    $\hat{j} \leftarrow j \in \Psi_{\text{passive}} : x_j \geq x_k \forall k \in \Psi_{\text{passive}}$
    $\Psi_{\text{active}} \leftarrow \{\hat{j}\} \cup \Psi_{\text{active}}$
    $\Psi_{\text{passive}} \leftarrow \Psi_{\text{passive}} \setminus \{\hat{j}\}$
    $\tilde{C}_1^*, c_{\Psi_{\text{active}}} \leftarrow$ EstimateDefect($\phi_{\Psi_{\text{active}}}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  return $s_{\Psi_{\text{active}}}$

OptimizeLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\phi_{\Lambda_D}, \tilde{C}_D, c_{\Psi_{\text{active}}} \leftarrow$ MutateLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\tilde{C}_D^{\text{stop}} \leftarrow C_{\text{stop}}(f_{\text{stringent}})^{D-1}$
  $m_{\text{reopt}} \leftarrow 0$
  while $\tilde{C}_D > \tilde{C}_D^{\text{stop}}$ **and** $m_{\text{reopt}} < M_{\text{reopt}}$
    $\{\xi_1, \ldots, \xi_{M_{\text{perturb}}}\}, \hat{\phi}_{\Lambda_D} \leftarrow$ WeightedMutationSampling($\phi_{\Lambda_D}, \tilde{C}_d^{\mathcal{T}}, M_{\text{perturb}}$)
    $\hat{\phi}_{\Lambda_D}, \hat{C}_D, \hat{c}_{\Psi_{\text{active}}} \leftarrow$ MutateLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
    if $\hat{C}_D < \tilde{C}_D$
      $\phi_{\Lambda_D}, \tilde{C}_D, c_{\Psi_{\text{active}}} \leftarrow \hat{\phi}_{\Lambda_D}, \hat{C}_D, \hat{c}_{\Psi_{\text{active}}}$
      $m_{\text{reopt}} \leftarrow 0$
    else
      $m_{\text{reopt}} \leftarrow m_{\text{reopt}} + 1$
  return $\phi_{\Lambda_D}, \tilde{C}_{\Lambda_D}$

MutateLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\tilde{C}_D, c_{\Psi_{\text{active}}} \leftarrow$ EstimateDefect($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  $\gamma_{\text{unfavorable}} \leftarrow \emptyset$
  $m_{\text{unfavorable}} \leftarrow 0$
  $\tilde{C}_D^{\text{stop}} \leftarrow C_{\text{stop}}(f_{\text{stringent}})^{D-1}$
  while $\tilde{C}_D > \tilde{C}_D^{\text{stop}}$ **and** $m_{\text{unfavorable}} < M_{\text{unfavorable}}$
    $\xi, \hat{\phi}_{\Lambda_D} \leftarrow$ WeightedMutationSampling($\phi_{\Lambda_D}, \{c_j^1, \ldots, c_j^{|\phi_j|} \forall j \in \Psi_{\text{active}}\}, 1$)
    if $\xi \in \gamma_{\text{unfavorable}}$
      $m_{\text{unfavorable}} \leftarrow m_{\text{unfavorable}} + 1$
    else
      $\hat{C}_D, \hat{c}_{\Psi_{\text{active}}} \leftarrow$ EstimateDefect($\hat{\phi}_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
      if $\hat{C}_D < \tilde{C}_D$
        $\phi_{\Lambda_D}, c_{\Psi_{\text{active}}} \leftarrow \hat{\phi}_{\Lambda_D}, \hat{c}_{\Psi_{\text{active}}}$
        $\tilde{C}_D \leftarrow \hat{C}_{C_D}$
        $m_{\text{unfavorable}} \leftarrow 0$
        $\gamma_{\text{unfavorable}} \leftarrow \emptyset$
      else
        $m_{\text{unfavorable}} \leftarrow m_{\text{unfavorable}} + 1$
        $\gamma_{\text{unfavorable}} \leftarrow \gamma_{\text{unfavorable}} \cup \{\xi\}$
  return $\phi_{\Lambda_D}, \tilde{C}_D, c_{\Lambda_D}$

EstimateDefect($\phi_{\Lambda_d}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  $Q_{\Lambda_d}, P_{\Lambda_d} \leftarrow$ NodalProperties(ϕ_{Λ_d})
  $\tilde{Q}_{\Psi_{\text{active}}}, P_{\Psi_{\text{active}}} \leftarrow$ ComplexProperties($Q_{\Lambda_d}, P_{\Lambda_d}$)
  $x_{\Psi^0}^0 = A_{\Psi^0,j} y_j\ \forall j \in \Psi_{\text{active}}$
  if $\Psi_{\text{passive}} \neq \emptyset$
    $x_{\Psi^0}^0 = x_{\Psi^0}^0(1 - f_{\text{stop}} f_{\text{passive}})$
  $\tilde{x}_{\Psi_{\text{active}}} \leftarrow$ ComplexConcentrations($\tilde{Q}_{\Psi_{\text{active}}}, x_{\Psi^0}^0$)
  $\tilde{n}_{\Psi_{\text{active}}} \leftarrow$ ComplexDefect($P_{\Psi_{\text{active}}}, s_{\Psi_{\text{active}}}$)
  $\tilde{c}_{\Psi_{\text{active}}} \leftarrow$ TestTubeDefect($n_{\Psi_{\text{active}}}, \tilde{x}_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  $\tilde{C}_d \leftarrow \sum c_{\Psi_{\text{active}}}$
  return $\tilde{C}_d, c_{\Psi_{\text{active}}}$

Algorithm 3.1: Pseudocode for hierarchical test tube ensemble defect optimization. For a given set of target secondary structures, s_Ψ, and target concentrations, y_Ψ, a set of designed sequences, ϕ_Ψ, is returned by the function call OptimizeTube($s_\Psi, y_\Psi, \Psi, \Psi_{\text{on}}, \Psi_{\text{off}}$).

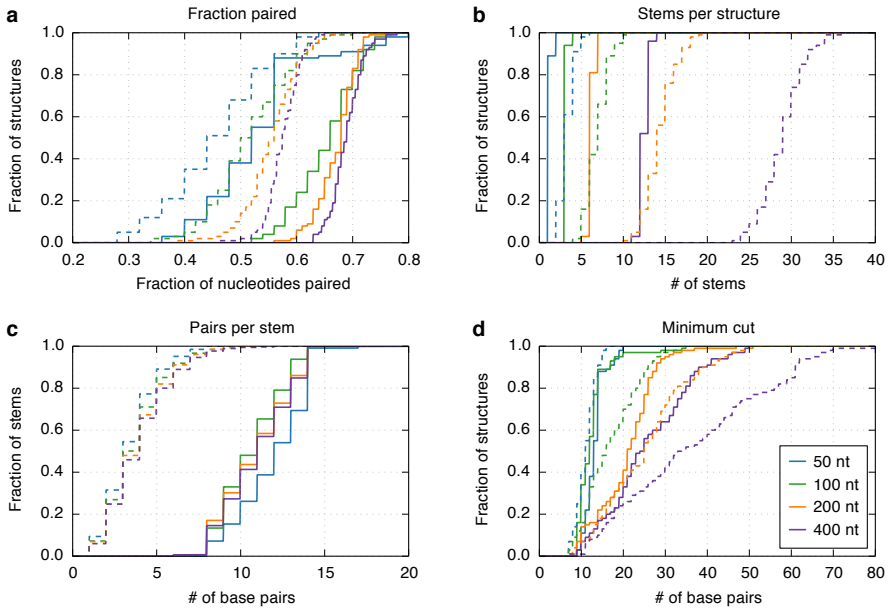


Figure 3.5: Structural features of the dimer target structures for the on-target complexes in the standard test set. a) The fraction of bases paired. b) The number of duplex stems per structure. c) The number base pairs per stem. d) The minimum number of base pairs that must be cut to disconnect a strand from a structure. Structures shown for the engineered test set (solid lines) and random test set (dashed lines).

set, each on-target structure was generated by calculating the minimum free energy structure of a random RNA dimer at 37 °C. Within each target test tube, there are two on-target dimers (each with a target concentration of 1 µM) and 106 off-target monomers, dimers, trimers, and tetramers (each with vanishing target concentrations), representing all complexes of up to $L_{\max} = 4$ strands (excluding the two on-target dimers). For each test set, fifty target test tubes were generated for each on-target dimer size, $|\phi| \in \{50, 100, 200, 400\}$ nt, with all strands the same length in each target test tube. The structural properties of the on-target structures in the engineered and random test sets are summarized in Figure 3.5. Typically, the random test set contains on-target structures with a lower fraction of paired nucleotides, more stems, and shorter stems (as short as one base pair). For the design studies that follow, new target test tubes were generated from scratch. The design algorithm was not tested on these target test tubes prior to generating the depicted results.

3.3.3 Sequence design trials

For all studies, five independent design trials were performed for each target test tube. Design trials were run on a cluster of 2.53 GHz Intel E5540 Xeon dual-processor/quad-core nodes with 24 GB of memory per node. Unless otherwise noted, trials were performed on a single computational core using the default algorithm parameters of Table 3.1. Design quality is plotted [35] as the normalized

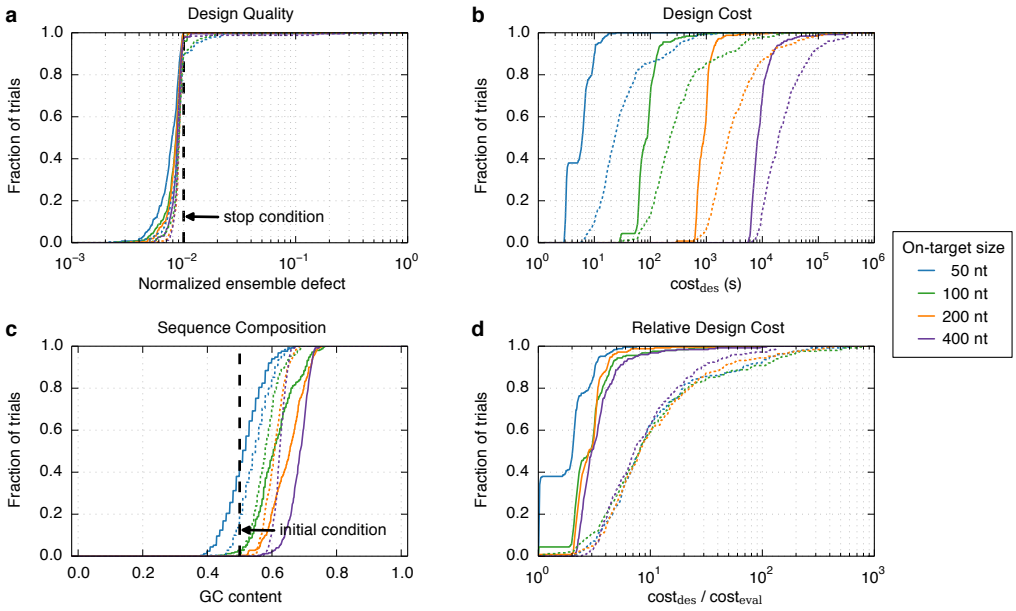


Figure 3.6: Algorithm performance for test tube design. a) Design quality. The stop condition is depicted as a dashed black line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed black line. d) Cost of sequence design relative to a single evaluation of the objective function. RNA design at 37 °C for the engineered test set (solid lines) and random test set (dashed lines).

test tube ensemble defect, C/y_{nt}. Data are typically presented as cumulative histograms over design trials. Our primary test scenario is RNA sequence design at 37 °C with $f_{\mathrm{stop}} = 0.01$ (i.e., less than 1% of the nucleotides should be incorrectly paired within the test tube at equilibrium) for the engineered test set.

3.4 Results and discussion

3.4.1 Algorithm performance for test tube design

Figure 3.6 demonstrates the performance of the test tube design algorithm on the engineered and random test sets. For each target test tube, the algorithm designs for on-target dimers (each with a target secondary structure and target concentration) and against 106 off-target monomers, dimers, trimers, and tetramers (each with vanishing target concentration). Most design trials surpass the desired design quality (normalized test tube ensemble defect ≤ 0.01; panel a). Typical design cost ranges from less than a second for test tubes with 50-nt on-targets in the engineered test set to approximately 3 hours for test tubes with 400-nt on-targets in the random test set (panel b). Starting from random initial sequences, the desired design quality can be achieved with a broad range of GC contents (panel c). The typical cost of design relative to the cost of a single evaluation

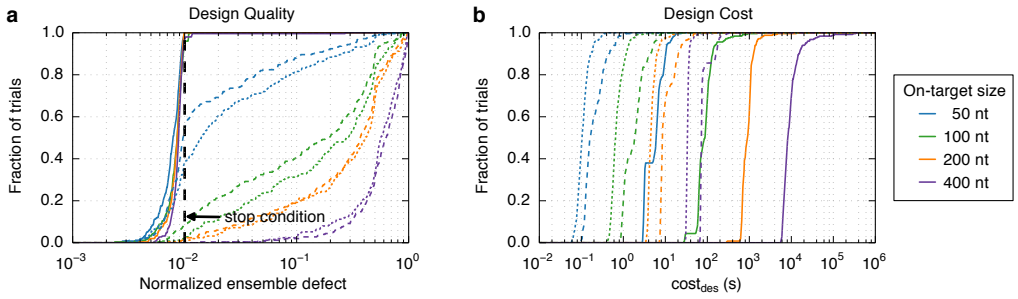


Figure 3.7: The importance of designing against off-target complexes. Comparison of design quality for test tube design performed using an ensemble containing all off-targets up to size $L_{\max} = 0$ (dotted line; $|\Psi_{\text{off}}| = 0$), $L_{\max} = 2$ (dashed line; $|\Psi_{\text{off}}| = 12$), or $L_{\max} = 4$ (solid line; $|\Psi_{\text{off}}| = 106$). Design quality is evaluated by calculating the test tube ensemble defect for a reference ensemble containing all off-targets up to size $L_{\max} = 5$. The stop condition is depicted as a dashed black line. RNA design at 37 °C for the on-targets in the engineered test set.

of the test tube ensemble defect is only a factor of 3 for the engineered test set and a factor of 10 for the random test set.

3.4.2 Importance of designing against off-targets

Is it important to include off-target complexes in the test tube ensemble so that the design algorithm can actively destabilize them? To examine this question in the context of the engineered test set, Figure 3.7 compares the design quality for sequences designed in a test tube ensemble containing either no off-targets (equivalent to complex design), all off-targets up to dimers, or all off-targets up to tetramers. The quality of the resulting design is evaluated using a reference test tube ensemble including all off-targets up to pentamers. Only the sequences designed against all off-targets up to tetramers are consistently of high quality; designing against no off-targets or against all off-targets up to dimers typically leads to very poor sequence designs.

We note that in the absence of off-target destabilization, the strands in an on-target complex will often also form a dimerized off-target complex (containing two copies of each strand) at non-negligible concentration. In anticipation of this phenomenon, we recommend actively designing against these dimerized off-targets, which was achieved for the engineered test set (containing two dimer on-targets per target test tube) by designing against all off-targets up to tetramers.

3.4.3 Contributions of algorithmic ingredients

To avoid the expense of evaluating candidate mutations on all off-target complexes throughout the design process, test tube ensemble focusing partitions the complexes in Ψ into the sets Ψ_{active} and Ψ_{passive}. To efficiently accept or reject each candidate mutation, the test tube ensemble defect is estimated at the leaf level of the decomposition forest obtained via hierarchical ensemble decomposition

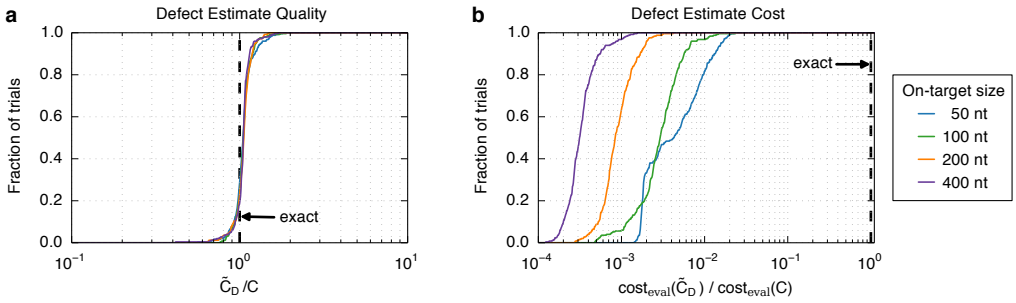


Figure 3.8: Performance of test tube ensemble defect estimation. a) Accuracy relative to exact defect. b) Cost relative to exact defect cost. The exact test tube ensemble defect, C, is calculated using all on- and off-target complexes $j \in \Psi$. The test tube ensemble defect estimate, $\tilde{C}_D$, is calculated using the leaf nodes, $k \in \Lambda_D$, of the final decomposition forest obtained by hierarchically decomposing the structural ensembles of the on- and off-target complexes $j \in \Psi_{\text{active}}$. RNA design at 37 °C for the engineered test set. For each design trial, comparisons are made using the final designed sequences, ϕ_Ψ.

of the complexes in Ψ_{active}. Figure 3.7 demonstrates that the estimated defect typically closely approximates the exact defect, but at a cost that is lower by 2-3 orders of magnitude for the engineered test set. Figure 3.9 demonstrates that the cost savings resulting from hierarchical decomposition become substantial as the size of the complexes in Ψ_{active} increases (due to the increasing depth of the decomposition forest), and that the cost savings resulting from test tube ensemble focusing are substantial independent of complex size (due to the large number of off-targets in Ψ_{passive}).

3.4.4 Robustness to model perturbations

Algorithms for the analysis and design of equilibrium nucleic acid secondary structure depend on empirical free energy models [40, 48, 62, 63, 69]. It is inevitable that the parameter sets in these models will continue to be refined, so it is important that assessments of design quality are robust to parameter perturbations. Figure 3.10 demonstrates that for the engineered test set, the predicted quality of most sequence designs is typically robust to 3% parameter perturbations (with the test tube ensemble defect often less than the stop condition), and even to 10% parameter perturbations (with test tube ensemble defect often within a factor of 2 of the stop condition), but not to 30% parameter perturbations (with test tube ensemble defect rarely within a factor of 10 of the stop condition).

3.4.5 Designing competing on-target complexes

In the engineered test set, each of the four strands appears in exactly one of two on-target dimers, so there is no disadvantage to stabilizing these dimers to the maximum extent possible, since the target concentration for all off-target complexes is zero. However, if there are multiple on-target

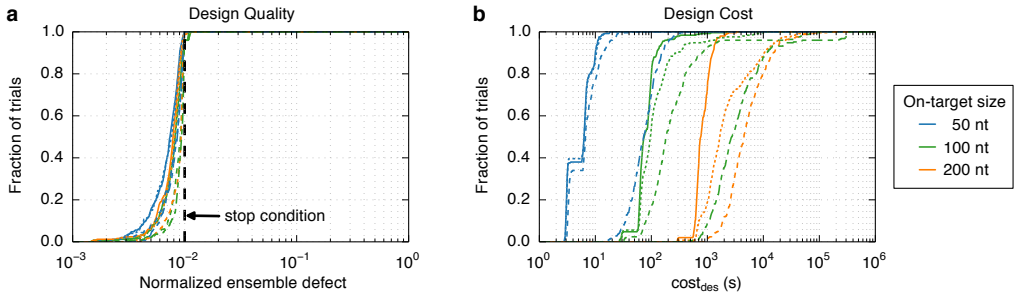


Figure 3.9: Efficiency implications of test tube ensemble focusing and hierarchical ensemble decomposition. a) Design quality. The stop condition is depicted as a dashed black line. b) Design cost. Comparison of test tube design performed with: the full algorithm (including test tube ensemble focusing and hierarchical ensemble decomposition permitting multiple exclusive split-points per parent; solid lines); test tube ensemble focusing and hierarchical ensemble decomposition permitting only a single split-point per parent (dotted lines); test tube ensemble focusing but no hierarchical ensemble decomposition (dashed lines); or no test tube partitioning and no hierarchical ensemble decomposition (uneven dashed lines) (Note: the 200-nt tests were prohibitively expensive for this test, so they were not completed).

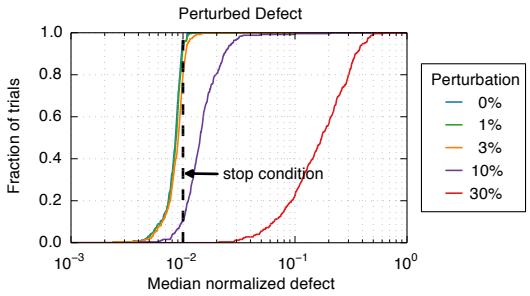


Figure 3.10: Robustness of design quality predictions to model perturbations. Each sequence design was evaluated using 100 perturbed physical models, with each parameter perturbed by Gaussian noise with a standard deviation of 0, 1, 3, 10, or 30 percent of the parameter modulus. RNA design at 37 °C for target test tubes in the standard test set with $|\phi| = 200$. The stop condition is depicted as a dashed black line.

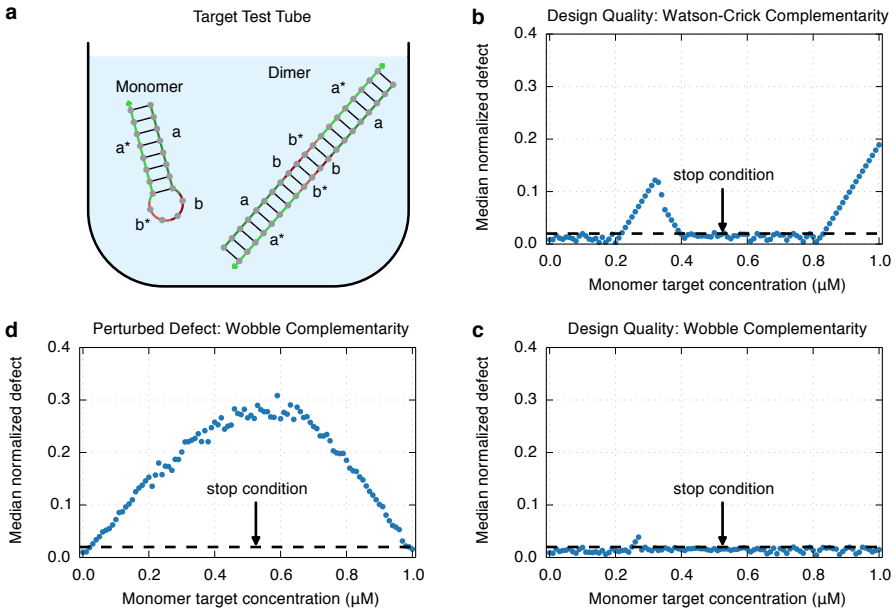


Figure 3.11: Test tube design with competing on-target copmlexes. A range of target test tubes were defined, with the monomer target concentration, y_{monomer}, ranging from 0 to 1 µM in 0.01 µM increments and the total strand concentration held fixed at 1 µM (i.e., $y_{\text{dimer}} = (1 - y_{\text{monomer}})/2$). b) Median design quality with Watson-Crick complementarity constraints. c) Median design quality with wobble complementarity constraints. d) Robustness of design quality predictions to perturbations in model parameters for sequence designs with wobble complementarity constraints. For each design trial, the median test tube ensemble defect was calculated over 100 perturbed physical models (each parameter perturbed by Gaussian noise with a standard deviation of 10 percent of the parameter modulus). RNA sequence design at 37 °C with $L_{\max} = 2$ (i.e. no off-targets). The test tube stop condition is depicted as a dashed black line ($f_{\text{stop}} = 0.02$).

complexes competing for the same strands, then the algorithm must balance the relative stability of these competing on-targets. To examine this challenge, we consider target test tubes in which a strand is intended to form both a monomer hairpin and a dimer duplex (Figure 3.11a), varying the target concentration of the monomer while keeping the total strand concentration fixed. Figure 3.11b demonstrates that typical design quality varies greatly depending on the target monomer concentration (i.e. depending on the desired relative stability of the monomer and dimer on-targets). For example, the algorithm typically succeeds in satisfying the stop condition for low monomer target concentrations, but struggles to satisfy the stop condition for high monomer target concentrations. These designs were performed requiring Watson–Crick complementarity constraints. If wobble pairs are permitted, typical design performance significantly improves (Figure 3.11c), reflecting the additional flexibility provided to the algorithm.

Because of the competition between on-target complexes it is interesting to revisit the question of robustness to model perturbations. The perturbation studies of Figure 3.11d demonstrate that the predicted design quality is typically robust to model perturbations for test tubes where one on-target dominates the other, but becomes more sensitive to model perturbations for test tubes where both on-targets are in competition at non-saturated target concentrations. Hence, for applications where on-targets are intended to form at non-saturated concentrations, it is more likely that the relative stabilities of the on-targets will need to be fine-tuned based on experimental measurements to account for imperfections in the physical model. Fortunately, many applications seek to saturate all on-targets at maximum concentration, reducing the sensitivity of computational predictions to perturbations in model parameters.

3.4.6 Test tube design with large numbers of on- and off-target complexes

Figure 3.12 demonstrates the performance of the algorithm for target test tubes containing a larger number of on- and off-target complexes. Typical design trials surpass the desired design quality (normalized test tube ensemble defect ≤ 0.01; panel a) and the typical design cost is less than four times the cost of a single evaluation of the test tube ensemble defect (panel d), ranging from 10 seconds for a test tube containing 1 on-target and 14 off-targets, to 8 hours for a test tube containing 8 on-targets and 17976 off-targets (panel b).

3.5 Conclusion

Test tube design provides a powerful framework for engineering nucleic acid base pairing. The desired equilibrium base pairing properties for the test tube are specified as an arbitrary number of on-target complexes, each with a target secondary structure and target concentration, and an arbitrary number of off-target complexes, each with vanishing target concentration. Given a target test

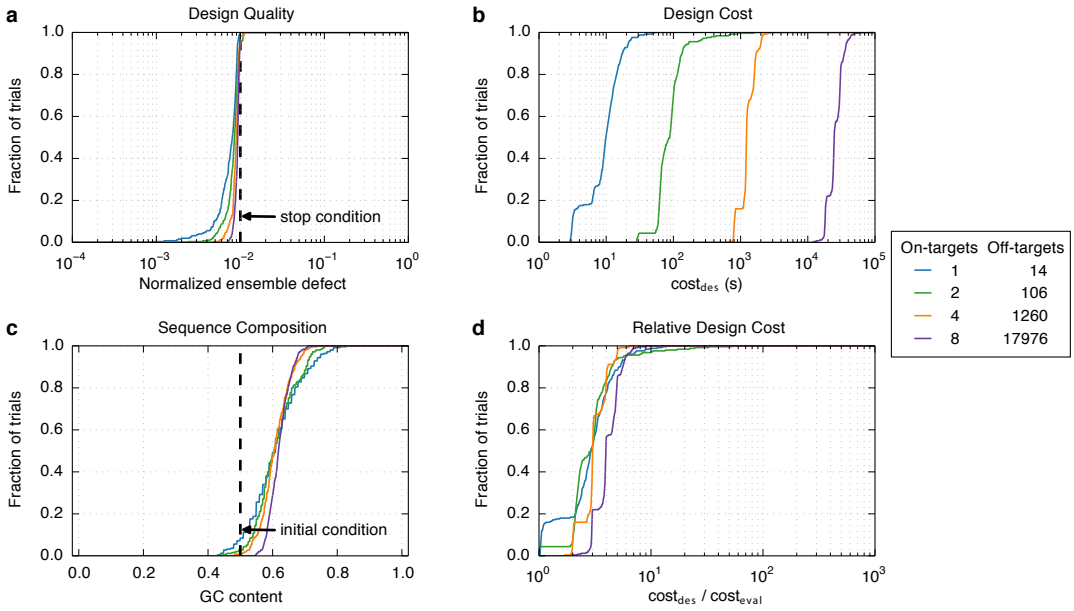


Figure 3.12: Test tube design with large numbers of on- and off-target complexes. Target test tubes contain $\Psi_{\text{on}} = 1$, 2, 4, or 8 on-target dimers and all off-target complexes up to size $L_{\max} = 4$ (corresponding to $|\Psi_{\text{off}}| = 14$, 106, 1260, or 17976 off-targets). a) Design quality. The stop condition is depicted as a dashed black line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed line. d) Cost of sequence design relative to a single evaluation of the test tube ensemble defect for a test tube containing all complexes up to size $L_{\max} = 4$. RNA sequence design at 37 °C. Fifty target test tubes for each value of Ψ_{on}; on-target dimers randomly selected from the subset of the engineered test set with $|\phi| = 100$ nt.

tube, the test tube ensemble defect quantifies the concentration of incorrectly paired nucleotides at equilibrium. Test tube ensemble defect optimization implements a positive design paradigm (stabilize on-targets) and a negative design paradigm (destabilize off-targets) at two levels: a) designing for the on-target structure and against the off-target structures within the structural ensemble of each on-target complex [15, 83], and b) designing for the on-target complexes and against the off-target complexes within the ensemble of the test tube. Both paradigms are crucial at both levels in order to achieve high-quality test tube designs with a low test tube ensemble defect. Test tube ensemble focusing and hierarchical ensemble decomposition enable efficient estimation and optimization of the test tube ensemble defect for target test tubes representative of design challenges in the molecular programming and synthetic biology communities, typically yielding sequences with a test tube ensemble defect $\leq 1\%$ at a design cost within an order of magnitude of the cost of a single evaluation of sequence quality.

3.6 Appendix

Appendix Section A.2 explains the algorithm used to generate structures for the engineered test set. Appendix B contains structural feature summaries for the engineered and random test sets, a description of the branch-and-bound algorithm used during hierarchical decomposition using multiple exclusive split points, additional studies on algorithm performance: complex design, random seed composition, design material, and parallel performance, sensitivity studies for the design parameters, and descriptions of the archive content for the test tube design algorithm.

Chapter 4

Design of nucleic acid reaction pathways via constrained thermodynamic optimization

We describe an algorithm for designing reaction pathways of interacting nucleic acid strands by designing multiple thermodynamic ensembles. Each ensemble optimizes a target structure or a target test tube and can represent a step within a reaction pathway. Sequence design is performed by optimizing a weighted sum of test tube ensemble defects. Hierarchical ensemble decomposition and test tube ensemble focusing enable efficient design of the combined objective. To satisfy experimentally important sequence constraints, we use a combinatorial constraint satisfaction algorithm to generate random initial sequences and mutations. This constrained, hierarchical sequence optimization algorithm makes it practical to perform thermodynamic optimization for systems of interest in the molecular programming and synthetic biology communities.

4.1 Introduction

The previous chapter describes an algorithm which generalizes the design of a single target structure to the design of a target test tube, designing the structure and concentration for a set of on-target complexes and designing against a set of off-target complexes. Conditional, dynamic processes move through multiple states. To design these systems, we formulate a generalization of test tube design. The new design algorithm optimizes a set of dilute solutions while satisfying combinatorial sequence constraints. The dilute solutions being optimized represent states in the reaction pathway.

The differences between this algorithm and the test tube algorithm are of three types. First, the optimization procedure is generalized to account for the presence of multiple design objectives. Second, a constraint satisfaction procedure is included in the design algorithm to enforce combinatorial constraints. Third, minor changes are made to structural ensemble decomposition to handle multiple target structures for the same complex. In the following sections, we describe the changes in the

formulation needed to handle multiple tubes and combinatorial sequence constraints. Section 4.2 describes more details about the design process, including the remaining changes.

4.1.1 Multistate design problem specification

A multistate design problem is specified as a set of target test tubes, $\mathcal{T}$. Each tube, $h \in \mathcal{T}$, contains a set of desired on-target complexes, Ψ^h_{on}, and a set of undesired off-target complexes, Ψ^h_{off}. The total set of all considered complexes is then

$$\Psi = \bigcup_{h \in \mathcal{T}} \Psi^h_{\text{on}} \cup \Psi^h_{\text{off}}.$$

This can be partitioned into complexes that appear as on-targets in at least one tube,

$$\Psi_{\text{on}} = \bigcup_{h \in \mathcal{T}} \Psi^h_{\text{on}},$$

and off-target complexes that appear as off-targets in at least one tube, but never appear as on-targets in any tube,

$$\Psi_{\text{off}} = \Psi \setminus \Psi_{\text{on}}.$$

Each complex, $j \in \Psi$, is specified as a strand ordering, π_j, corresponding to structural ensemble $\Gamma(\pi_j)$. For each tube, h, the user specifies a target secondary structure, $s_{h,j}$, and target concentration, $y_{h,j}$, for all complexes $j \in \Psi^h_{\text{on}}$. The target concentration is vanishing for all other complexes $j \in \Psi^h_{\text{off}}$. It is convenient to specify the maximum off-target size $L_{\max}$ for each tube, including all complexes with no more than $L_{\max}$ strands as off-targets in Ψ.

To capture these design goals, we seek to perform sequence optimization on sequences ϕ_Ψ such that each test tube has a test tube ensemble defect $C(\phi_{\Psi^h}, s_{\Psi^h}, y_{\Psi^h})$ that satisfies its test tube stop condition,

$$C(\phi_{\Psi^h}, s_{\Psi^h}, y_{\Psi^h}) \leq C^h_{\text{stop}}, \tag{4.1}$$

where C^h_{stop} is the allowed concentration of nucleotides in the incorrect state in the test tube. This is the product of a user specified stop condition for the tube, f^h_{stop}, and the total concentration of nucleotides in the tube, y^h_{nt},

$$C^h_{\text{stop}} = f^h_{\text{stop}} y^h_{\text{nt}}.$$

When the design problem is a single tube with a single on-target complex, this is equivalent to complex ensemble defect optimization [83]. When the design problem is a single test tube, this equivalent to test tube ensemble defect optimization as described in the previous chapter. When the design problem contains multiple tubes, each with a single on-target complex, this is equivalent

to ensemble defect optimization for multiple target structures [81]. The multistate design algorithm is thus a generalization of all three optimization problems.

4.1.2 Constraints

Nucleic acid designs in synthetic biology and molecular programming typically have constraints on the sequence space. These are often used to specify interactions (or lack thereof) with nucleic acids, proteins, ligands, or other external entities that are not captured in the thermodynamic specification. In this section, we motivate and define the available sequence constraints.

The constraint satisfaction problem is defined over the set of variables, v, their domains, $\mathcal{D}$, and constraints over subsets of variables, $\mathcal{C}$. Each unique sequence position, ϕ^a, is linked to a variable, v^b, that takes a value in domain $\mathcal{D}^b = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{U}\}$. A constraint, $\mathcal{C}_i$, is defined over a list of variables,

$$v_{\mathcal{C}_i} \equiv \left(v^1_{\mathcal{C}_i}, \ldots, v^{|v_{\mathcal{C}_i}|}_{\mathcal{C}_i}\right) : v^a_{\mathcal{C}_i} \in v$$

called the scope of $\mathcal{C}_i$. Each constraint is defined by a subset of the Cartesian product of the domains of the variables in its scope

$$\mathcal{C}_i \equiv v_{\mathcal{C}_i} \mapsto\subseteq \mathcal{D}^1_{\mathcal{C}_i} \times \ldots \times \mathcal{D}^{|v_{\mathcal{C}_i}|}_{\mathcal{C}_i}.$$

For example, Watson-Crick complementarity can be defined between two variables (v^a, v^b). The constraint defines the set of possible joint assignments for the two variables

$$\mathcal{C}^{\mathrm{WC}}_i \equiv (v^a, v^b) \mapsto \{(\mathtt{A}, \mathtt{U}), (\mathtt{C}, \mathtt{G}), (\mathtt{G}, \mathtt{C}), (\mathtt{U}, \mathtt{A})\}; \tag{4.2}$$

a subset of all sixteen possible assignments between unconstrained nucleotides. A partial instantiation,

$$V \equiv \{v^a \mapsto d \in \mathcal{D}^a\},$$

maps a subset of variables to values within their respective domains. A partial instantiation is consistent with constraint $\mathcal{C}_i$ if every variable in its scope, $v_{\mathcal{C}_i}$, is instantiated and all instantiated values are found within a single tuple in the constraint. A complete instantiation is an instantiation that maps all variables to values within their respective domains.

Before describing each of the constraints, it is useful to define the sequence distance d^{seq} between a list of nucleotides q and a degenerate pattern r as the number of nucleotides q^a that are not instantiations of the corresponding degenerate nucleotide r^a,

$$d^{\mathrm{seq}}(q, r) = \sum_{a \in 1, \ldots, |q|} \begin{cases} 1 : q^a \in r^a \\ 0 : q^a \notin r^a \end{cases}, \tag{4.3}$$

e.g., $d^{\text{seq}}(ACGU, \texttt{SSWW}) = 2$.

In the remainder of this section we define the available sequence constraints.

- **Identical** constraints are defined between two variables v^a and v^b. An identical constraint is consistent with any instantiation that maps both variables to the same value. We write this:

$$\mathcal{C}_i^{\text{ident}} \equiv (v^a, v^b) \mapsto \{(\texttt{A}, \texttt{A}), (\texttt{C}, \texttt{C}), (\texttt{G}, \texttt{G}), (\texttt{U}, \texttt{U})\}. \tag{4.4}$$

  These constraints are implicitly specified through the use of sequence domains and explicitly defined through the use of the `identical` statement.

- **Complementary** constraints are also defined between two variables, v^a and v^b. Complementarity has two variants, Watson-Crick complementarity and wobble complementarity. Watson-Crick complementarity is defined by (4.2). Wobble complementarity also allows wobble [G·U] pairs:

$$\mathcal{C}_i^{\text{wobble}} \equiv (v^a, v^b) \mapsto \{(\texttt{A}, \texttt{U}), (\texttt{C}, \texttt{G}), (\texttt{G}, \texttt{C}), (\texttt{G}, \texttt{U}), (\texttt{U}, \texttt{A}), (\texttt{U}, \texttt{G})\}. \tag{4.5}$$

  These constraints are implicitly specified through the use of sequence domains and by the base pairing of target structures, and explicitly defined through the use of the `complement` statement.

- **Sequence assignment** constraints act on a single variable and limit it to a subset of its domain. These are typically specified using the generalized nucleotides listed in Table 2.1. For example, a constraint $\mathcal{C}_i^{\texttt{W}}$ on variable v^a is defined by

$$\mathcal{C}_i^{\texttt{W}} \equiv (v^a) \mapsto \{(\texttt{A}), (\texttt{U})\}.$$

  These are explicitly defined during sequence domain definition using the `domain` statement.

- **Match** constraints act on a set of variables. Each match constraint specifies the sequence distance between a given pattern and the constraint scope using a minimum, $f_i^{\min}$, and maximum, $f_i^{\max}$, match fraction.

$$\mathcal{C}_i^{\text{match}} \equiv v_{\mathcal{C}_i^{\text{match}}} \mapsto \left\{ q : f_i^{\min} \leq \frac{d(q,r)}{|q|} \leq f_i^{\max} \right\}. \tag{4.6}$$

  If $f_i^{\min} = f_i^{\max} = 1$, this is equivalent to a sequence assignment constraint over a set of variables. These are explicitly defined using the `match` statement.

- **Library** constraints restrict a list of nucleotides to match enumerated sets of allowed patterns $\{r\}_i$. Each library constraint allows an engineer to enumerate a set of allowed domain sequences. The library constraint is defined by the set of possible sequences that exactly match

any pattern in the library

$$\mathcal{C}_i^{\text{library}} \equiv v_{\mathcal{C}_i^{\text{library}}} \mapsto \{q : \exists r \in \{r\}_i \text{ s.t. } d^{\text{seq}}(q, r) = 0\}. \tag{4.7}$$

This constraint can be used for many purposes. If part of a mechanism is a protein coding sequence, a user can specify codon constraints. If a user has a pre-existing library of orthogonal toeholds, this constraint can force the algorithm to optimize among them for a particular design. These constraints are explicitly defined using the `library` and `libseq` statements.

- **External sequence** constraints are a special case of the library constraint. We define external sequence constraints as library constraints where the set of allowed patterns $\{r\}_i$ is defined by the set of all subsequences of a source pattern, r_i^{source}, of a given length, $|v_{\mathcal{C}_i^{\text{ext}}}|$. This can be used to conveniently specify interactions with long targets, e.g., target mRNA sequences. These are explicitly defined through the use of the `source` and `window` statements.

- **Pattern prevention** constraints avoid overly repetitive sequences. Some repetitive sequences are known to form alternative secondary structures that may not be captured by the nearest neighbor model [27, 48, 61]. To forbid these, pattern prevention constraints, $\mathcal{C}_i^{\text{pattern}}$, can be specified. For a list of variables, $v_{\mathcal{C}_i^{\text{pattern}}}$, and pattern r, the prevented pattern constraint defines the set of assignments such that no subsequence $q^{a:a+|r|-1}$ matches r:

$$\mathcal{C}_i^{\text{pattern}} \equiv \left\{ q : \nexists q^{i:i+|r|-1} \text{ s.t. } d^{\text{seq}}(q^{i:i+|r|-1}, r) = 0 \right\}. \tag{4.8}$$

These are explicitly defined through the use of `prevent` statements.

Finding sequences that satisfy these constraints is NP-hard in general. Empirically, however, the constraints used for nucleic acid design in the molecular programming community are practical to solve. We use a randomized depth-first branch and propagate algorithm to find random sequences and random mutations that satisfy all the constraints during sequence initialization and sequence mutation [13], as described later.

4.2 Algorithm

We describe a multistate design algorithm based on optimizing a scaled sum of test tube ensemble defects. This multistate defect is reduced via iterative mutation of a random initial sequences. On-target decomposition of target structures yields a forest of decomposition trees. Candidate mutations are evaluated efficiently by estimating test tube ensemble defects based on nodal properties calculated at the leaves of the decomposition forest. During leaf optimization, defect-weighted mutation sampling is used to select a nucleotide to mutate with probability proportional to its contribution

to the estimated multistate defect. Each mutation is chosen to satisfy combinatorial constraints using a branch and propagate procedure. After leaf optimization, parental defects are evaluated based on the optimized sequences of the children. As optimized subsequences are merged toward the root level of the forest, decomposition defects that arise due to crosstalk between subsequences are eliminated via redecomposition and reoptimization of parents that are not well approximated by their children. After subsequences are merged to the root level, the full defect for each tube is calculated for the first time, including all on- and off-target complexes in all test tube ensembles. Any off-target complexes that form at appreciable concentrations in any target tube are decomposed and added to the decomposition forest and actively destabilized during subsequent forest reoptimization. The elements of this hierarchical sequence design algorithm are described below and detailed in the pseudocode of Algorithm 4.1.

We take the approach of optimizing a weighted sum of objectives, relying on the user to specify meaningful stop conditions for each objective. For this purpose, we define the *multistate defect contribution* as the ensemble defect normalized by its allowed defect

$$\mathcal{M}^h = \frac{C^h}{C^h_{\text{stop}}}. \tag{4.9}$$

The multistate defect is defined as the average multistate defect contribution over all tubes

$$\mathcal{M} = \frac{1}{|\mathcal{T}|} \sum_{h \in \mathcal{T}} \mathcal{M}^h. \tag{4.10}$$

To avoid over-optimization of objectives that satisfy their stop condition, we define the *thresholded multistate defect* as the average multistate defect, with each contribution thresholded to unity

$$\mathcal{M}^{\text{thresh}} = \frac{1}{|\mathcal{T}|} \sum_{h \in \mathcal{T}} \max(\mathcal{M}^h, 1). \tag{4.11}$$

This quantity captures the defect in excess of each stop condition, normalized by the stringency of the stop condition. When all tubes satisfy their respective stop conditions, the thresholded multistate defect is minimized at $\mathcal{M}^{\text{thresh}} = 1$.

4.2.1 Ensemble focusing

To reduce the cost of sequence optimization, the set of complexes, Ψ, is partitioned into two disjoint sets:

$$\Psi = \Psi_{\text{active}} \cup \Psi_{\text{passive}} \tag{4.12}$$

$$\emptyset = \Psi_{\text{active}} \cap \Psi_{\text{passive}} \tag{4.13}$$

where Ψ_{active} denotes complexes that will be actively designed, and Ψ_{passive} denotes complexes that will inherit sequence information from Ψ_{active}. Initially, we set

$$\Psi_{\text{active}} = \Psi_{\text{on}} \tag{4.14}$$

so that only structures that are on-targets in at least one tube are initially actively designed. The user-specified on-target structures provide the basis for hierarchical ensemble decomposition, which enables efficient sequence design. The sequences for complexes $j \in \Psi_{\text{active}}$ are initialized using a randomized branch and bound algorithm, as described below.

4.2.2 Hierarchical ensemble decomposition

Exact evaluation of the test tube ensembles, C^h, requires calculation of the defect contributions, $c_{h,j}$, for each complex, $j \in \Psi_{\text{active}}$. To reduce this cost during sequence optimization, we seek to estimate these contributions at lower cost by hierarchically decomposing each actively designed complex into a tree of subensembles, yielding a forest of $|\Psi_{\text{active}}|$ decomposition trees. Hierarchical ensemble decomposition proceeds similarly to Section 3.2.6 except that multiple target structures can exist for a single complex. This requires minimal modification to the previously described procedures.

First, we define the set of structure matrices at the root of the decomposition tree for complex j:

$$\{S\}_k = \{S(s_{h,j})\ \forall h \in \mathcal{T}\}. \tag{4.15}$$

Following the notation from Section 3.2.8, we redefine structure-guided decomposition (previously defined using (3.27)) to use a set of exclusive split points, $\{F\}$, to allow the algorithm to decompose multiple target structures for a single node.

$$\tilde{B}(\{S\}_k) \equiv \left\{ \{F\} : \begin{array}{c} |\{S\}_k| = \sum_{S_h \in \{S\}_k} \sum_{F_i \in \{F\}} \min_{a \cdot b \in F_i^{\pm}} S_h^{a,b} \\ \min_{\{F_i\}}(|\phi_{k_{l_i}}|, |\phi_{k_{r_i}}|) \geq N_{\text{split}} \\ F_i \otimes F_j \ \ \forall F_i \neq F_j \in \{F\} \end{array} \right\}, \tag{4.16}$$

and optimal set of exclusive split points,

$$\{\tilde{F}\}^* \equiv \min_{\{F\} \in \tilde{B}(\{S\}_k)} \sum_{F_i \in \{F\}} \left(|\phi_{k_{l_i}}|^3 + |\phi_{k_{r_i}}|^3 \right). \tag{4.17}$$

Similarly, the set of possible split points for structure- and probability-guided decomposition (pre-

viously defined in (3.32)) is redefined to account for the possibility of multiple target structures:

$$\hat{B}(\{S_k\}, P_k) \equiv \left\{ \{F\} : \begin{array}{c} \{F\} = \{F\}_i \cup \{F\}_j, \{F\}_i \in \tilde{B}(\{S_k\}), \{F\}_j \in \bar{B}(P_k) \\ F_i \otimes F_j \quad \forall F_i \neq F_j \in \{F\} \end{array} \right\},$$

and the optimal set of exclusive split points,

$$\{F\}^* \equiv \min_{\{F\} \in \hat{B}(\{S\}_k, P_k)} \sum_{F_i \in \{F\}} \left(|\phi_{k_{l_i}}|^3 + |\phi_{k_{r_i}}|^3 \right). \tag{4.18}$$

Both of these quantities are equivalent to those presented in Section 3.2.8 if the complex has a single target structure.

As in test tube design, let Λ denote the set of all nodes in the forest. Let Λ_d denote the set of all nodes at depth d. Let $\Lambda_{d,j}$ denote the set of all nodes at depth d resulting from decomposition of complex j. Note that each complex contributes a single tree to the decomposition forest, even when the complex appears in multiple tubes.

4.2.2.1 Stop condition stringency

To account for a basal level of crosstalk moving back up the tree, we make the stop condition more strict as we go deeper in the forest:

$$\tilde{C}^{h,d}_{\text{stop}} = C^h_{\text{stop}} (f_{\text{stringent}})^{d-1}. \tag{4.19}$$

To capture this in the thresholded multistate defect during optimization, we make the threshold more strict, but not the normalization. That is, we define the thresholded multistate defect estimate at level d as

$$\mathcal{M}^{\text{thresh}}_d \equiv \frac{1}{|\mathcal{T}|} \sum_{h \in \mathcal{T}} \frac{\max(\tilde{C}^{h,d}, C^{h,d}_{\text{stop}})}{C^h_{\text{stop}}}. \tag{4.20}$$

This quantity can take values below unity. When the level stop conditions (4.19) are satisfied for all tubes, h, at level d, the thresholded multistate defect estimate takes its minimal value of $(f_{\text{stringent}})^{d-1}$.

4.2.3 Multistate defect estimate from nodal contributions

To estimate the multistate defect at low cost, we must calculate partition function and pair probability estimates of the root ensemble using information calculated efficiently at descendant nodes in the forest. Using these, we can estimate the concentrations, complex ensemble defects, and test tube ensemble defect contributions for each complex in each tube.

The partition functions, $\tilde{Q}_j$, and pair probability matrices, $\tilde{P}_j$, for each complex, $j \in \Psi_{\text{active}}$, are estimated as described in Section 3.2.4.1 and 3.2.4.3, respectively. For each complex, j, we then calculate its estimated concentration, $x_{h,j}$, for each tube, h, in which it appears using deflated strand concentrations as described in Section 3.2.4.2. Similarly, we estimate the complex ensemble defect for each on-target j in tube h:

$$\tilde{n}_{h,j} = |\phi_j| - \sum_{\substack{1 \le a \le |\phi_j| \\ 1 \le b \le |\phi_j|+1}} \tilde{P}_j^{a,b} S(s_{h,j}). \tag{4.21}$$

The concentration and complex ensemble defect are combined to produce a test tube ensemble defect contribution estimate for each tube,

$$\tilde{c}_{h,j} = \tilde{n}_{h,j} \min\left(\tilde{x}_{h,j}, y_{h,j}\right) + |\phi_j| \max\left(y_{h,j} - \tilde{x}_{h,j}, 0\right), \tag{4.22}$$

and the test tube ensemble defect, which is the sum of these contributions

$$\tilde{C}^{h,d} = \sum_{j \in \Psi_{\text{on}}^h} \tilde{c}_{h,j} \tag{4.23}$$

4.2.4 Leaf mutation

To reduce computational cost, all candidate mutations are evaluated at the leaf nodes, $k \in \Lambda_D$, of the decomposition forest. Leaf mutation attempts to reduce the leaf-level thresholded multistate defect estimate $\tilde{\mathcal{M}}_d^{\text{thresh}}$ until all tubes, h, satisfy their *leaf stop conditions*

$$\tilde{C}^{h,D} \le C_{\text{stop}}^{h,D}. \tag{4.24}$$

A candidate mutation set is accepted if it decreases the thresholded multistate defect estimate and rejected otherwise.

We perform *defect weighted mutation sampling* by selecting nucleotide a for mutation with probability proportional to the contribution of nucleotide a to the stop-normalized ensemble defect in each unsatisfied objective. After choosing a nucleotide position to mutate, the resulting instantiation is chosen from all other feasible instantiations at that position. The remaining nucleotides in candidate sequence $\hat{\phi}$ are instantiated using a randomized branch and propagate algorithm that preferentially chooses branches that minimize the number of mutated nucleotides compared to the previous sequence. The changed nucleotides are characterized by the mutation set ξ (a list of changed instantiations).

A candidate sequence, $\hat{\phi}_{\Lambda_D}$, is evaluated via calculation of the multistate defect estimate, $\tilde{\mathcal{M}}_D$, if the candidate mutation, ξ, is not in the set of previously rejected mutations, $\gamma_{\text{unfavorable}}$. The set,

$\gamma_{\text{unfavorable}}$, is updated after each unsuccessful mutation and cleared after each successful mutation. The counter $m_{\text{unfavorable}}$ is used to keep track of the number of consecutive failed mutation attempts; it is incremented after each unsuccessful mutation and reset to zero after each successful mutation. Leaf mutation terminates unsuccessfully if $m_{\text{unfavorable}} \geq M_{\text{unfavorable}}$. The outcome of leaf mutation is the set of leaf sequences, ϕ_{Λ_D}, corresponding to the lowest encountered $\tilde{C}_D$.

4.2.5 Leaf reoptimization

After leaf mutation terminates, if any leaf stop conditions are unsatisfied, leaf reoptimization commences. During each round of leaf reoptimization, the algorithm perturbs the current sequence by reseeding M_{reseed} distinct sequence positions using defect weighted mutation sampling. Mutation locations are chosen without replacement, based on the defect calculated over the sequences returned from leaf mutation. Mutations are chosen sequentially using the randomized branch and bound procedure used in leaf mutation. The resulting perturbed sequences are used as the initial sequences for the new round of leaf mutation. The reoptimized candidate sequences, $\hat{\phi}_{\Lambda_D}$, are accepted if they decrease $\tilde{\mathcal{M}}_D^{\text{thresh}}$ and rejected otherwise. The counter m_{reopt} keeps track of the number of consecutive rejections of reoptimized sequences. It is incremented after each rejection and reset to zero after each acceptance. Leaf reoptimization terminates successfully if all leaf stop conditions are satisfied, and unsuccessfully if $m_{\text{reopt}} \geq M_{\text{reopt}}$. The outcome of leaf reoptimization is the set of leaf sequences, ϕ_{Λ_D}, corresponding to the lowest encountered $\tilde{\mathcal{M}}_D^{\text{thresh}}$.

4.2.6 Subsequence merging, redecomposition, and reoptimization

After leaf reoptimization terminates, parent nodes at depth $d = D - 1$ merge their left and right child sequences to create the set of candidate sequences $\hat{\phi}_{\Lambda_d}$. The parental defect estimate, $\tilde{\mathcal{M}}_d^{\text{thresh}}$, is calculated and candidate sequences, $\hat{\phi}_{\Lambda_d}$, are accepted if they decrease $\tilde{\mathcal{M}}_d^{\text{thresh}}$, and are rejected otherwise. If all *parental stop conditions*:

$$\tilde{C}^{h,d} \leq \max(C_{\text{stop}}^{h,d}, \tilde{C}^{h,d+1}/f_{\text{stringent}}) \tag{4.25}$$

are satisfied, merging continues up to the next level in the forest. Otherwise, failure to satisfy the parental stop condition indicates the existence of *decomposition defects* resulting from low probability base pairs surrounding a split point, invalidating the ensemble decomposition. For each tube that fails to satisfy (4.25), the parent node that exhibits the largest ensemble defect reduction when replaced by its children is chosen for structure- and probability-guided ensemble decomposition. Additional parents are redecomposed until

$$\tilde{C}^{h,d} - \tilde{C}^{h,d+1*}/f_{\text{stringent}} \leq f_{\text{redecomp}}(\tilde{C}^{h,d} - \tilde{C}^{h,d+1}/f_{\text{stringent}}) \tag{4.26}$$

is satisfied for all tubes that failed to satisfy their parental stop condition. Here, $\tilde{C}^{h,d+1}$ is the child defect estimate before any redecomposition, $\tilde{C}^{h,d+1*}$ is the child defect after redecomposition, and $f_{\text{stringent}} \in (0,1)$. If a parent is chosen for redecomposition and its split points are still eligible, those split points are forbidden and will not be chosen in any future decompositions.

The current sequences at depth d are then pushed down to all nodes below depth d, and a new round of leaf mutation and leaf reoptimization is performed. Following this, merging begins again. Subsequence merging and reoptimization terminates successfully when the parental stop condition (4.25) is satisfied at depth $d = 1$. The outcome of subsequence merging, redecomposition, and reoptimization is the set of subsequences $\phi_{\Psi_{\text{active}}}$, corresponding to the lowest encountered $\tilde{\mathcal{M}}_1$.

4.2.7 Test tube evaluation, refocusing, and reoptimization

Initial forest optimization is performed for the on-target complexes in Ψ_{active}, neglecting complexes that are off-targets in all tubes, Ψ_{passive}. At the termination of forest optimization, the thresholded multistate defect estimate at depth $d = 1$, $\mathcal{M}_1^{\text{thresh}}$, is calculated. For this estimate, the test tube defect estimates are based on estimated concentrations, $\tilde{x}_{\Psi^h_{\text{active}}}$, calculated using deflated total strand concentrations (3.12) to create a built-in defect allowance for the effect of the neglected off-target in Ψ^h_{passive}. For the first time, the full thresholded multistate defect, $\mathcal{M}^{\text{thresh}}$, is then calculated, including all complexes in Ψ. For this calculation, all test tube defects are based on complex concentrations calculated using the full strand concentrations (3.11).

Sequence design terminates successfully if all test tube ensemble defects satisfy their corresponding test tube stop condition (4.1), or are no greater than the forest-estimated defect (3.18):

$$C^h \leq \max\bigl(C^h_{\text{stop}}, \tilde{C}^{h,1}\bigr). \tag{4.27}$$

This condition allows sequence design to terminate if the actual defect contributions resulting from the off-target complexes in Ψ_{passive} are no greater than the built-in defect allowances resulting from deflation of the total strand concentrations during forest optimization. If (4.27) is unsatisfied, this implies the existence of a focusing defect,

$$C^h - \tilde{C}^{h,1} > 0, \tag{4.28}$$

for some tube, $h \in \mathcal{T}$. For each tube, h, that fails to satisfy (4.27), the off-target complex, $j \in \Psi_{\text{passive}}$, with the highest concentration in the tube is transferred to Ψ_{active} and decomposed using probability-guided decomposition. This is repeated until the refocusing stop condition,

$$C^h - \tilde{C}^{h,1*} \leq f_{\text{refocus}}(C^h - \tilde{C}^{h,1}), \tag{4.29}$$

is satisfied for all tubes that failed to satisfy (4.27). Here, $\tilde{C}^{h,1}$ is the estimated defect before augmenting Ψ_{active}, and $\tilde{C}^{h,1*}$ is the newly estimated defect using the augmented Ψ_{active}. Finally, the decomposition forest is reinitialized to include the newly decomposed complexes and forest optimization recommences, designing against the formation of the newly included off-targets. This process of tube refocusing is repeated until all stop conditions are satisfied. This is guaranteed to occur after adding all complexes to Ψ_{active}, i.e., $\Psi_{\text{active}} = \Psi$.

4.2.8 Constraint solving

During sequence initialization and sequence mutation, the algorithm needs to find a set of sequences that satisfy the sequence constraints. We first describe the branch and propagate algorithm to find mutations and then handle sequence initialization as a special case. To find feasible mutated sequences, we use a branch and propagate algorithm, as described in Chapter 5 of Dechter [13]. For each mutation, the previous sequences, ϕ, and a position to mutate, a, are used in the mutation generation procedure. A new instantiation for position a is chosen from its domain, then branch and propagate commences. At each branching step in the algorithm, an uninstantiated variable, $\hat{v}_b$, is chosen at random, weighted by the size of its domain. The implications for each feasible value are explored, and the one that implies the fewest mutations with respect to the original sequence, ϕ, is chosen for a subsequent branching step. Constraints are propagated using arc consistency. The same process is used during sequence initialization, but no nucleotide instantiation is initially chosen, and no previous sequence is defined; the instantiation of all nucleotides is done at random. The result of constraint solving is a new candidate sequence $\hat{\phi}$ that satisfies all specified constraints. If no feasible sequences exist, the algorithm instead returns a warning.

4.2.9 Structural defect weighting

The desired behavior of a nucleic acid system may be strongly dependent on the folding of a particular region of a target structure, but only weakly dependent on other regions. To allow engineers to differentially weight nucleotide defects in different regions of a target structure, we define structural defect weights $\alpha^a_{h,j}$ as a scaling factor for the nucleotide defect of complex j at nucleotide a in tube h. Each weight must be in the range $[0, 1]$. The contribution of nucleotide $a \in \phi_j$ to the weighted complex ensemble defect for any tube, h, is

$$\bar{n}^a_j = \alpha^a_{h,j} n^a_{h,j}.$$

OptimizeTubes($s_{\mathcal{T},\Psi}, y_{\mathcal{T},\Psi}, \Psi, \Psi_{\text{on}}, \Psi_{\text{off}}, \mathcal{T}$)
  $\phi_\Psi \leftarrow$ InitSeq($\emptyset, s_{\mathcal{T},\Psi}, \Psi$)
  $\Psi_{\text{active}}, \Psi_{\text{passive}} \leftarrow \Psi_{\text{on}}, \Psi_{\text{off}}$
  $\phi_\Lambda, s_\Lambda, \Lambda, D \leftarrow$ Decompose($\phi_{\Psi_{\text{active}}}$, $s_{\Psi_{\text{active}}}$)
  $\phi_\Psi, \tilde{C}^{\mathcal{T},1} \leftarrow$ OptimizeForest($\phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $C^{\mathcal{T}}, \mathcal{M}^{\text{thresh}} \leftarrow$ EvaluateDefects($\phi_\Psi, s_\Psi, y_\Psi$)
  $\hat{\phi}_\Psi, \hat{C}^{\mathcal{T}}, \hat{\mathcal{M}}^{\text{thresh}} \leftarrow \phi_\Psi, C^{\mathcal{T}}, \mathcal{M}^{\text{thresh}}$
  while $\exists h \in \mathcal{T} : \hat{C}^h > \max(C^h_{\text{stop}}, \tilde{C}^{h,1})$
    $s_{\Psi_{\text{active}}} \leftarrow$ AugmentActive($s_{\Psi_{\text{active}}}$, $y_\Psi, \hat{C}^{\mathcal{T}}, \tilde{C}^{\mathcal{T},1}, \hat{\phi}_\Psi$)
    $\phi_\Lambda, s_{\Psi_{\text{active}}}, \Lambda, D \leftarrow$ Decompose($\phi_{\Psi_{\text{active}}}$, $s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
    $\hat{\phi}_\Psi, \tilde{C}^{\mathcal{T},1} \leftarrow$ OptimizeForest($\phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
    $\hat{C}^{\mathcal{T},1}, \hat{\mathcal{M}}^{\text{thresh}} \leftarrow$ EvaluateDefects($\hat{\phi}_\Psi, s_\Psi, y_\Psi$)
    if $\hat{\mathcal{M}}^{\text{thresh}} < \mathcal{M}^{\text{thresh}}$
      $\mathcal{M}^{\text{thresh}}, \phi_\Psi \leftarrow \hat{\mathcal{M}}^{\text{thresh}}, \hat{\phi}_\Psi$
  return ϕ_Ψ

OptimizeForest($\phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\tilde{M}^{\text{thresh}}_{1,\ldots,D} \leftarrow \infty$
  $C^{h,d}_{\text{stop}} \leftarrow C^h_{\text{stop}} (f_{\text{stringent}})^{d-1}\ \forall h \in \mathcal{T}, \forall d \in 1, \ldots, D$
  pstop $\leftarrow$ **false**
  while $\neg$pstop
    $\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},D} \leftarrow$ OptimizeLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
    $d \leftarrow D - 1$
    pstop $\leftarrow$ **true**
    while $d \geq 1$ **and** pstop
      $\hat{\phi}_{\Lambda_d} \leftarrow$ MergeSeq($\phi_{\Lambda_{d+1}}$)
      $\hat{C}^{\mathcal{T},d}, \hat{\mathcal{M}}^{\text{thresh}}_d \leftarrow$ EstimateDefect($\hat{\phi}_{\Lambda_d}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
      if $\hat{\mathcal{M}}^{\text{thresh}}_d < \tilde{\mathcal{M}}^{\text{thresh}}_d$
        $\phi_{\Lambda_d}, \tilde{\mathcal{M}}^{\text{thresh}}_d \leftarrow \hat{\phi}_{\Lambda_d}, \hat{\mathcal{M}}^{\text{thresh}}_d$
      if $\exists h \in \mathcal{T} : \tilde{C}^{h,d} > \max\left[C^{h,d}_{\text{stop}}, \tilde{C}^{h,d+1}/f_{\text{stringent}}\right]$
        pstop $\leftarrow$**false**
        $\Lambda, \hat{\phi}_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}} \leftarrow$ Redecompose($\Lambda, \phi_\Lambda, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
        for $d' = d+1, \ldots, D$
          $\phi_{\Lambda_{d'}} \leftarrow$ SplitSeq($\hat{\phi}_{\Lambda_{d'-1}}$)
          $\tilde{\mathcal{M}}_{d'} \leftarrow \infty$
      $d \leftarrow d - 1$
  return $\hat{\phi}_\Psi, \tilde{C}^{\mathcal{T},1}$

AugmentActive($s_{\Psi_{\text{active}}}, y_\Psi, C^{\mathcal{T}}, \tilde{C}^{\mathcal{T},1}, \phi_\Psi$)
  $C^{\mathcal{T}*,1} \leftarrow \tilde{C}^{\mathcal{T},1}$
  while $\exists h \in \mathcal{T} : C^h - \tilde{C}^{h*,1} > f_{\text{refocus}}(C^h - \tilde{C}^{h,1})$
    $\hat{h} \leftarrow h \in \mathcal{T} : C^h - \tilde{C}^{h*,1} \geq C^{h'} - \tilde{C}^{h',1*} \forall h' \in \mathcal{T}$
    $\hat{j} \leftarrow j \in \Psi_{\text{passive}} : x^{\hat{h}}_j \geq x^{\hat{h}}_k \forall k \in \Psi_{\text{passive}}$
    $\Psi_{\text{active}} \leftarrow \Psi_{\text{active}}, \cup\{\hat{j}\}$
    $\Psi_{\text{passive}} \leftarrow \Psi_{\text{passive}} \setminus \{\hat{j}\}$
    $\tilde{C}^{\mathcal{T},1}, \mathcal{M}^{\text{thresh}} \leftarrow$ EstimateDefects($\phi_{\Psi_{\text{active}}}$, $s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  return $\Psi_{\text{active}}, \Psi_{\text{passive}}$

OptimizeLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},D}, \tilde{\mathcal{M}}^{\text{thresh}}_D \leftarrow$ MutateLeaves(ϕ_{Λ_D}, $s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $m_{\text{reopt}} \leftarrow 0$
  while $m_{\text{reopt}} < M_{\text{reopt}}$ **and** $\exists h \in \mathcal{T} : \tilde{C}^{h,D} > C^{h,D}_{\text{stop}}$
    $\{\xi_1, \ldots, \xi_{M_{\text{perturb}}}\}, \hat{\phi}_{\Lambda_D} \leftarrow$ WeightedMutationSampling($\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},D}, M_{\text{perturb}}$)
    $\hat{\phi}_{\Lambda_D}, \hat{C}^{\mathcal{T},D}, \hat{\mathcal{M}}^{\text{thresh}}_D \leftarrow$ MutateLeaves(ϕ_{Λ_D}, $s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
    if $\hat{\mathcal{M}}^{\text{thresh}}_D < \tilde{\mathcal{M}}^{\text{thresh}}_D$
      $\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},D}, \mathcal{M}^{\text{thresh}}_D \leftarrow \hat{\phi}_{\Lambda_D}, \hat{C}^{\mathcal{T},D}, \hat{\mathcal{M}}^{\text{thresh}}_D$
      $m_{\text{reopt}} \leftarrow 0$
    else
      $m_{\text{reopt}} \leftarrow m_{\text{reopt}} + 1$
  return $\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},D}$

MutateLeaves($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, D$)
  $\tilde{C}^{\mathcal{T},D}, \tilde{\mathcal{M}}_D \leftarrow$ EstimateDefect($\phi_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
  $\gamma_{\text{unfavorable}} \leftarrow \emptyset$
  $m_{\text{unfavorable}} \leftarrow 0$
  while $\exists h \in \mathcal{T} : \tilde{C}^{h,D} > C^{h,D}_{\text{stop}}$ **and** $m_{\text{unfavorable}} < M_{\text{unfavorable}}$
    $\{\xi\}, \hat{\phi}_{\Lambda_D} \leftarrow$ WeightedMutationSampling($\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},d}, 1$)
    if $\xi \in \gamma_{\text{unfavorable}}$
      $m_{\text{unfavorable}} \leftarrow m_{\text{unfavorable}} + 1$
    else
      $\hat{C}^{\mathcal{T},D}, \hat{\mathcal{M}}^{\text{thresh}}_D \leftarrow$ EstimateDefect($\hat{\phi}_{\Lambda_D}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}$)
      if $\hat{\mathcal{M}}_D < \tilde{\mathcal{M}}_D$
        $\phi_{\Lambda_D}, \tilde{\mathcal{M}}_D, \tilde{C}^{\mathcal{T},D} \leftarrow \hat{\phi}_{\Lambda_D}, \hat{\mathcal{M}}_D, \hat{C}^{\mathcal{T},D}$
        $m_{\text{unfavorable}} \leftarrow 0$
        $\gamma_{\text{unfavorable}} \leftarrow \emptyset$
      else
        $m_{\text{unfavorable}} \leftarrow m_{\text{unfavorable}} + 1$
        $\gamma_{\text{unfavorable}} \leftarrow \gamma_{\text{unfavorable}} \cup \{\xi\}$
  return $\phi_{\Lambda_D}, \tilde{C}^{\mathcal{T},D}$

EstimateDefect($\phi_{\Lambda_d}, s_{\Psi_{\text{active}}}, y_{\Psi_{\text{active}}}, \mathcal{T}$)
  $Q_{\Lambda_d}, P_{\Lambda_d} \leftarrow$ NodalProperties(ϕ_{Λ_d})
  $\tilde{Q}_{\Psi_{\text{active}}}, \tilde{P}_{\Psi_{\text{active}}} \leftarrow$ ComplexProperties($Q_{\Lambda_d}, P_{\Lambda_d}$)
  for $h \in \mathcal{T}$
    $x^0_{\Psi^0} = A_{\Psi^0,j} y^h_j\ \forall j \in \Psi_{\text{active}}$
    if $\Psi^h_{\text{off}} \cap \Psi_{\text{passive}} \neq \emptyset$
      $x^0_{\Psi^0} = x^0_{\Psi^0}(1 - f_{\text{stop}} f_{\text{passive}})$
    $\tilde{x}_{\Psi^h_{\text{active}}} \leftarrow$ ComplexConcentrations($\tilde{Q}_{\Psi^h_{\text{active}}}, x^0_{\Psi^0}$)
    $\tilde{n}_{h,\Psi_{\text{active}}} \leftarrow$ ComplexDefects($s_{h,\Psi_{\text{active}}}, P_{\Psi_{\text{active}}}, Q_{\Psi_{\text{active}}}$)
    $\tilde{c}_{h,\Psi_{\text{active}}} \leftarrow$ TubeDefects($\tilde{n}_{h,\Psi_{\text{active}}}, \tilde{x}_{h,\Psi_{\text{active}}}, y_{h,\Psi_{\text{active}}}$)
    $\tilde{C}^{h,d} \leftarrow \sum \tilde{c}_{h,j}$
  $\mathcal{M}^{\text{thresh}}_d \leftarrow \frac{1}{|\mathcal{T}|} \sum_{h \in \mathcal{T}} \max(\tilde{C}^{h,d}/C^h_{\text{stop}}, (f_{\text{stringent}})^{d-1})$
  return $\tilde{C}^{\mathcal{T},d}, \mathcal{M}^{\text{thresh}}_d$

Algorithm 4.1: Multiobjective design. For a given set of target secondary structures, s_Ψ, target test tubes, $\mathcal{T}$, and target concentrations within each test tube, y_Ψ, a set of designed sequences, ϕ_Ψ, is returned by the function call OptimizeTubes($s_{\mathcal{T},\Psi}, y_{\mathcal{T},\Psi}, \Psi, \Psi_{\text{on}}, \Psi_{\text{off}}$).

If all weights are unity, this is equivalent to the complex ensemble defect. The analogous weighted complex ensemble defect estimates are defined equivalently

$$\tilde{\tilde{n}}_j^a = \alpha_{h,j}^a \tilde{n}_{h,j}^a$$

The weighted complex ensemble defect and the weighted complex ensemble defect estimate replace their unweighted versions everywhere in the optimization algorithm. The default weights are all unity.

4.3 Methods

The pathway design algorithm was evaluated on two groups of test problems. The first set of problems compares the performance of the pathway design algorithm to previously described algorithms for the special cases of single complex design and test tube design. The second set of problems are drawn from the molecular programming community and characterized with several types of sequence constraints.

4.3.1 Single complex and single test tube designs

The first group of test sets compare the performance of the pathway design algorithm to the Zadeh single-complex design algorithm [83] and to the test tube design algorithm described in Chapter 3.

These test sets were generated using the same test set as in the previous chapter. The engineered test set was produced by randomly combining helices and unpaired regions to produce target structures of the desired length. The length of the helices and unpaired regions were chosen to be consistent with feature lengths seen in the nucleic acid design literature. The random test set was chosen by generating random sequences, calculating the MFE structure for each sequence, and using that as a target structure. Every structure in both test sets was required to be connected by at least 8 base pairs to ensure feasibility during design.

4.3.2 Pathway designs

The second set of designs was created based on five real-world systems. Starting states, intermediate states, and final states were used as objectives. To explore the ability of the algorithm to design independently acting reaction pathways, five versions of each system were created, with $1, \ldots, 5$ copies of the system in each design specification, each designed to act independently. For example, five HCR designs were created, the smallest containing a single HCR system, and the largest containing five HCR systems designed to act orthogonally, that is, they should not interact with high affinity in the test tube. To implement orthogonality, select inputs, intermediates, and outputs from different

systems were designed in combined test tube ensembles, optimizing against off-target complexes that form due to cross-talk between the systems. Orthogonal sets were designed for all five system types, resulting in twenty-five design types for each test. For each design type, ten design trials were executed per condition. In this section, we describe the system types and rationales for the objective choices. It is informative to compare the reaction pathways specifications in Chapter 2 to the tube specifications below to see how each pathway is divided into distinct thermodynamic ensembles which can be optimized independently.

Figure 4.1 shows the design objectives used to optimize HCR. The first two tubes, 'Initiators' and 'Hairpins', optimize against interactions between the two initiators and the two hairpins from the same system, respectively. Tube 'Detect 1' optimizes for the binding of hairpin H1 with its initiator I1 to form intermediate D1, with an exposed initiation site for subsequent H2 addition. The H2 addition step is captured in 'Detect 2' by including the exposed tail of the previous intermediate, I2, with the second hairpin, H2, which should form the tail of the growing polymer after one more addition, D2. These two steps approximately capture the energetics of each subsequent addition. The final tube type, 'Cross-target', optimizes against off-target binding and activation. Each set of hairpins is designed in the context of all off-target initiators, designing against all possible dimers. This optimizes against initiators from independent systems binding to each other as well against HCR initiation by an off-target initiator.

The cooperative gate designs follow similar logic, as shown in Figure 4.2. Each input gate is optimized to form along with the downstream reporter, even in the presence of one or the other input. The inputs are designed to avoid interaction with each other. When both are present along with gate D, they should displace strand P, which releases the fluorophore-labeled F from the reporter gate, if present. One cross-target tube optimizes against interaction of the inputs T1 and T2 from one system with the input strands, gates, and reporter gates from other systems. The other tube optimizes against intermediate strand P from one system interacting with reporter gate R from another system. This system requires balancing the strength of tohold binding; binding must be strong to drive the reaction at experimental concentrations when both inputs are present, but weak enough to avoid persistent binding of either input to the gate in the absence of the other input.

Figure 4.3 shows the design specification for the boolean logic gates [65]. The 3-stranded gate molecule is designed in isolation, and the two input strands are designed together to be unstructured. The first step and output step are optimized in independent tubes. Cross-activation is captured in two sets of tubes; the cross-active tubes prevent inputs from binding to the wrong gate molecule, and the 'All Inputs' and 'All Gates' tubes prevent binding between independent input and gate molecules. Note that each tube that primarily optimizes a multi-stranded on-target complex ('Gate', 'Step' and 'Output' tubes) contains all species at a low concentration, 1 nM, to enforce high affinity. The remaining tubes design primarily against unintended binding. These contain all species at a

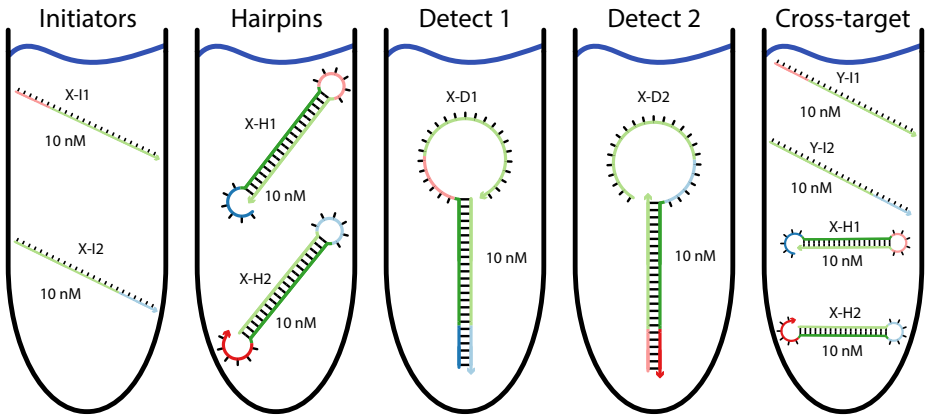


Figure 4.1: Design objectives for HCR [17, 11]. This design takes advantage of the symmetry of the addition steps, designing the end of the polymer after each addition step by creating a virtual initiator I2 in addition to the normal initiator I1. The 'Initiators' tube optimizes initiators to be single-stranded. The 'Hairpins' tube optimizes metastability of the hairpins in solution. The 'Detect 1' and 'Detect 2' tubes optimize the two addition events for an HCR polymer. The first optimizes the addition of H1 to a polymer or the initiator, I1, and the second optimizes H2 addition to the exposed tail of H1. The 'Cross-target' tube includes one set of hairpins, H1 and H2, from a system, X, and the initiators from all other systems, Y, and designs against all possible dimers. This tube avoids cross-activation of system X by any system Y and binding between the initiators of different systems. Each tube contains all off-targets containing up to $L_{\max} = 2$ strands. Designed using DNA at 25 °C.

higher concentration, 100 nM, to stringently avoid cross-talk.

Figure 4.4 shows the design specification for the catalytic assembly of three-arm junctions. The three hairpin binding steps are captured in independent 'Step' tubes, capturing the same symmetry as in the HCR designs. The final product is designed in a separate 'Product' tube. Each pairwise dimerization of hairpins is designed against in the 'Spurious' tubes. The cross-target tubes design against dimerization of any off-target input or intermediate hairpin tail with the each hairpin.

Figure 4.5 shows the design specification for the conditional Dicer substrates [33]. The reactant dimer and hairpin are designed to be stable when mixed in a dilute solution, even at a concentration of 100 nM. The two steps of the reaction are captured using the tubes Step 1 and Step 2. Unintended interactions between the input and hairpin are designed against. Cross-talk between orthogonal systems is designed against using the Cross-target tube. For this system, most nucleotides are constrained to be subsequences of either a detection target or a silencing target. The input molecule is constrained via an external sequence constraint to be a subsequence of the detection target, eGFP. The output duplex, consisting of the first 19 nucleotides of the dicer substrate, starting at the side with a 2-nt overhang, is completely independent of the input and is constrained via another external sequence constraint to be a subsequence of the silencing target dsRed2. These constraints reduce the number of feasible sequences for the design by over ten orders of magnitude compared to the

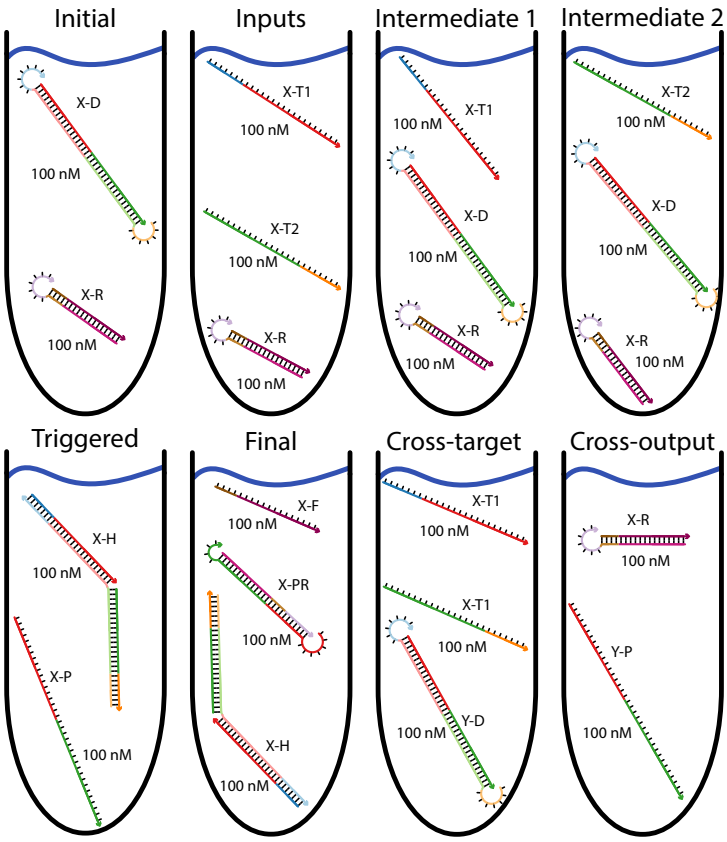


Figure 4.2: Design objectives for cooperative gates [84]. The first two tubes capture the desired initial states, optimizing against premature activation of the reporter RF (tube 'Initial') and against any dimerization of the input strands (tube 'Inputs'). The tubes 'Intermediate 1' and 'Intermediate 2' optimize against activation of the gate when only one input is present. The 'Triggered' tube is representative of the state of the tube with both inputs but without the reporter, and the intermediate output P is released. The 'Final' tube is representative of the final state of the gate when both inputs are present, and the final output F is released. The 'Cross-target' tube optimizes against cross-talk between the gate from one system, X, and the input strands from all other systems Y. The 'Cross-output' tube optimizes against cross-talk between the reporter from one system, X, and all possible intermediate outputs from the other systems, Y. Each tube contains all off-targets containing up to $L_{\max} = 3$ strands. Designed using DNA at 25 °C.

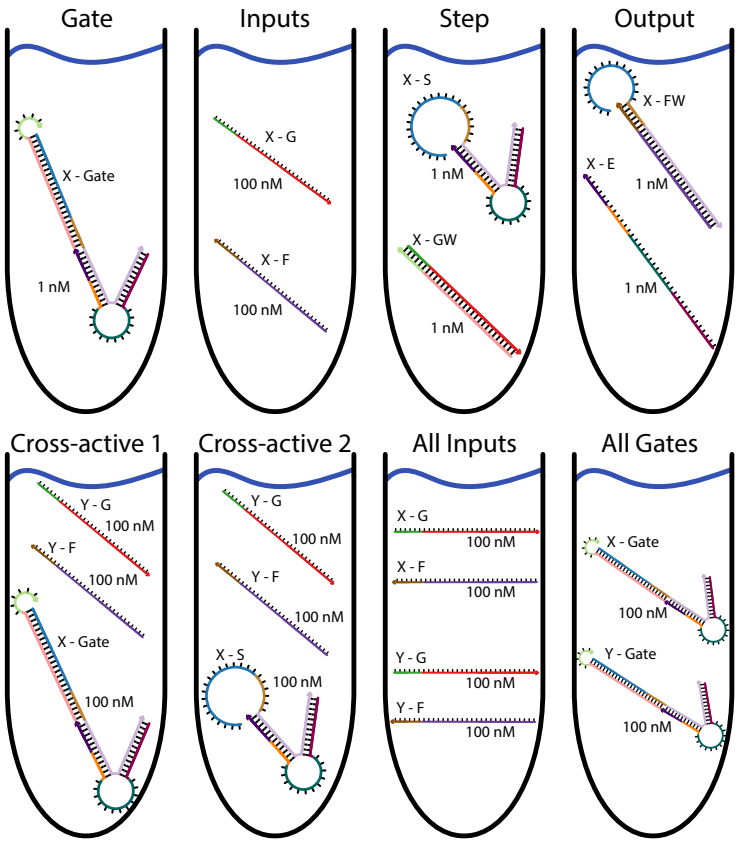


Figure 4.3: Design objectives for AND gates [65]. The first two tubes, 'Gate' and 'Inputs' capture desired initial states, optimizing against input strand dimerization and towards formation of the AND gate trimer. The 'Step' tube optimizes the first step of the reaction (after adding input strand G). The 'Output' tube optimizes for correct output production after adding the second input strand. The 'Cross-active 1' tube designs against interaction between the gate complex from one system, X, and the inputs from all other systems, Y. The 'Cross-active 2' tube designs against interactions between the intermediate complex from one system, X, and the inputs from all other systems, Y. In both of the 'Cross-active' tubes, all off-target input strands are included in the same tube as the gate or intermediate. For the design of three orthogonal AND gate systems, for instance, there would be three 'Cross-active 1' tubes, and each tube would include a single AND gate and four off-target input strands. The 'All Inputs' and 'All Gates' tubes include all input strands and all gates in the same dilute solution, designing against unintended binding of input strands and crosstalk between gate complexes. The 'Gate', 'Step', 'Output', 'Cross-active 1', and 'Cross-active 2' tubes contain all off-targets containng up to $L_{\max} = 3$ strands. The remaining tubes contain all off-targets containing up to $L_{\max} = 2$ strands. Designed using DNA at 25 °C.

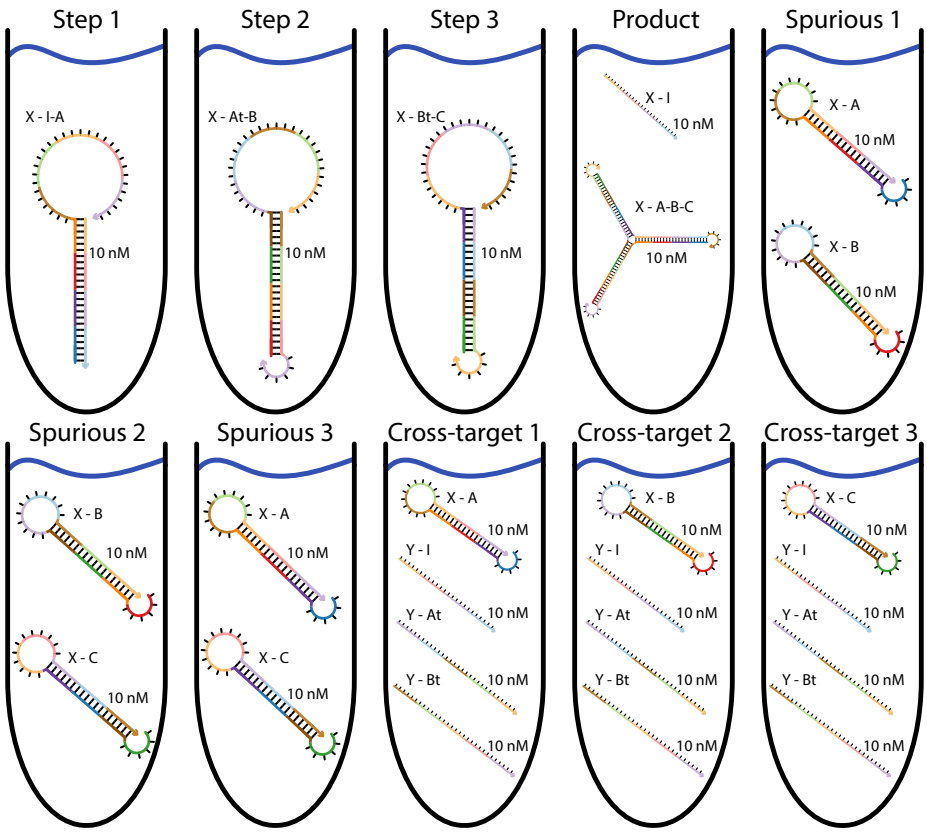


Figure 4.4: Design objectives for catalytic three-arm junction assembly [79]. This design takes advantage of the pattern of the three addition steps, including only the exposed tail of the previous hairpin in each addition step. The first three tubes, 'Step 1', 'Step 2', and 'Step 3', represent the three hairpin additions. Product formation is optimized in the 'Product' tube. 'Spurious 1', 'Spurious 2', and 'Spurious 3' optimize against all pairwise dimerizations between the hairpins. We cannot directly optimize against the three-arm junction forming since the mechanism is metastable. If the 'Product' tube forms correctly, the stable state of the three hairpins in solution will be the three-arm junction. Each 'Cross-target' tube optimizes against binding of the hairpin from one system, X, to the input strands and exposed tails of intermediates from the remaining systems, Y. For example, in the design of three orthogonal three-arm junction assembly instantiations, there are three tubes of the type Cross-target 1. Each of these tubes contains seven on-targets, one hairpin, and six single-stranded molecules, representing the input catalyst strand and exposed tails of hairpins from each of the other two systems. Each tube contains off-targets with up to $L_{\max} = 2$ strands except the 'Product' tube, which contains all off-targets with up $L_{\max} = 3$ strands. Designed using DNA at 25 °C.

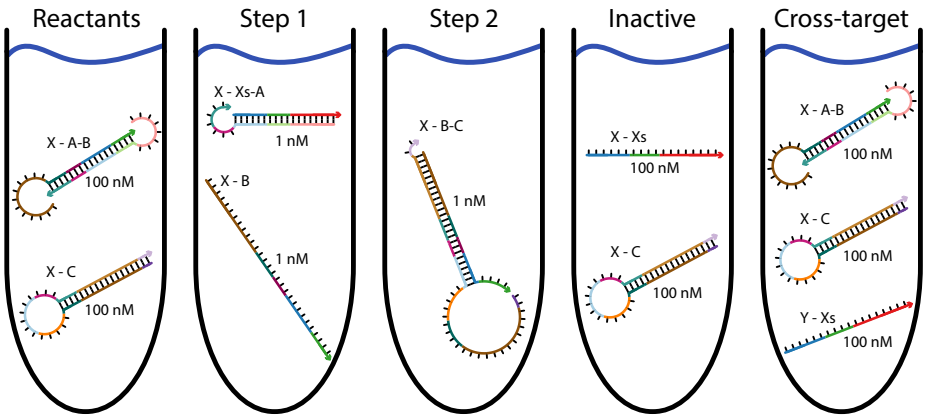


Figure 4.5: Design objectives for conditional Dicer [33]. The mechanism is captured in the first three tubes. The Reactants tube ensures the two input molecules are stable. The 'Step 1' tube optimizes the first step of the reaction: binding of A-B to input window Xs to form intermediate B and waste Xs-A. The 'Step 2' optimizes B binding to hairpin C, forming dimer B-C. This dimer is an RNA duplex that can trigger Dicer, silencing the knockout target gene. The 'Inactive' tube designs against unintended interactions between C and the input window Xs. The 'Cross-target' tube designs against off-target interactions between orthogonal systems. The sequence of each Xs strand is constrained to be a subsequences of the input target, eGFP, using an external sequence constraint. The sequence of the first 19 nucleotides of the duplex B-C was similarly constrained to be a subsequence of the knockdown target, dsRed2, using another external sequence constraint. Each tube contains all off-targets containing up to $L_{\max} = 3$ strands. Designed using RNA at 37 °C.

unconstrained design problem.

The number of tubes, on-targets, and off-targets for each design type are shown in Table 4.1. The design specification with the smallest number of complexes, a single HCR instantiation, contains 6 on-target structures and 6 off-target structures in a total of 5 tubes. The design specification with the largest number of complexes, 5 orthogonal cooperative gates, contains 40 on-target complexes and 3245 off-target complexes.

4.3.3 Implementation

The test tube design algorithm is coded in the C and C++ programming languages. The algorithm is available for non-commercial research purposes as part of the NUPACK web application and code base (`www.nupack.org`).

All designs were generated using Python scripts and encoded using the NUPACK design language. A specification of the design language is available in Appendix Section C.2. The Python scripts and resulting NUPACK design scripts are available in the supplementary archive file.

System type	# instantiations	$\lvert\mathcal{T}\rvert$	$\lvert\Psi_{\text{on}}\rvert$	$\lvert\Psi_{\text{off}}\rvert$
HCR	1	5	6	6
	2	10	12	20
	3	15	18	54
	4	20	24	96
	5	25	30	150
Cooperative gate	1	8	8	95
	2	16	16	318
	3	24	24	846
	4	30	32	1790
	5	36	40	3245
AND gate	1	8	7	58
	2	14	14	201
	3	20	21	549
	4	26	28	1190
	5	32	35	2180
3-arm junction	1	10	11	41
	2	20	22	116
	3	30	33	225
	4	40	44	368
	5	50	55	545
Conditional Dicer	1	5	6	32
	2	10	12	94
	3	15	18	213
	4	20	24	406
	5	25	30	690

Table 4.1: Number of objectives, on-targets, and off-targets for each design type. The number of instantiations is the number of orthogonal instantiations of the specified design type being designed.

4.4 Results

4.4.1 Special-case comparisons to previous algorithms

This algorithm generalizes complex ensemble defect optimization and test tube ensemble defect optimization using similar decomposition approaches. To ensure parity with these past algorithms for the special case of complex ensemble defect optimization, we tested all three algorithms on an engineered test set without designing against any off-target complexes. The results shown in Figure 4.6 demonstrate similar performance between the three algorithms for the special case of single-complex design. The 1% stop conditions are achieved by all algorithms (panel a). The test tube design algorithm and multistate design algorithm typically overshoot the stop condition by a smaller amount. The design cost is comparable between the algorithms. For intermediate lengths, the test tube and multistate design algorithms are typically slightly faster. The GC content is also comparable (panel c). All three design algorithms typically succeed in designing 400-nt complexes in just over 4/3 the cost of evaluating the structural ensemble defect for the sequence (panel d).

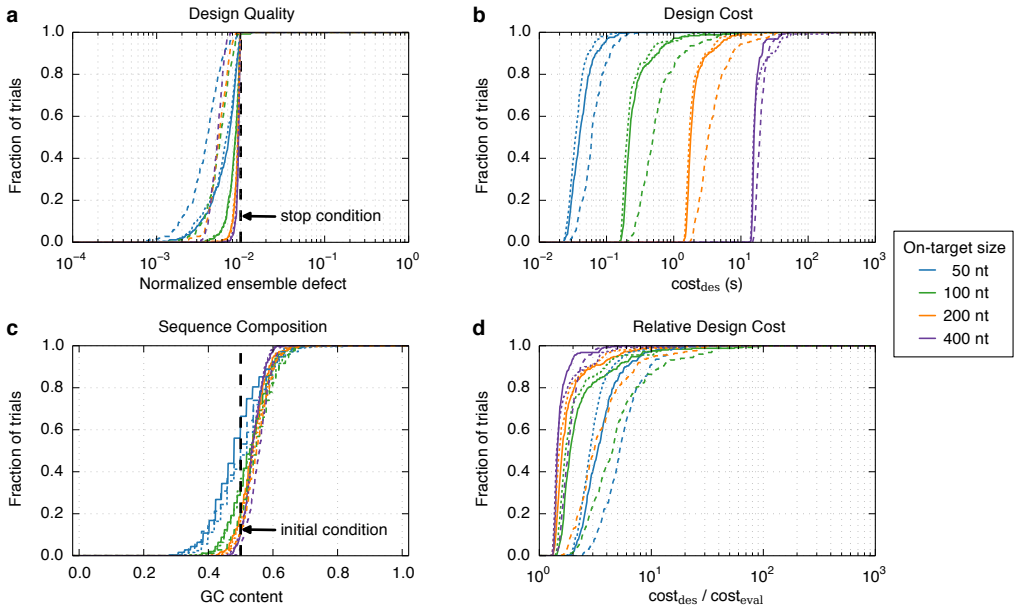


Figure 4.6: Multistate algorithm performance for single complex design. Algorithm performance for test tube design. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed line. d) Cost of sequence design relative to a single evaluation of the objective function. RNA design at 37 °C for the engineered test set for the Zadeh 2011 algorithm (dashed lines), the test tube design algorithm (dotted lines), and the multistate design algorithm (solid lines).

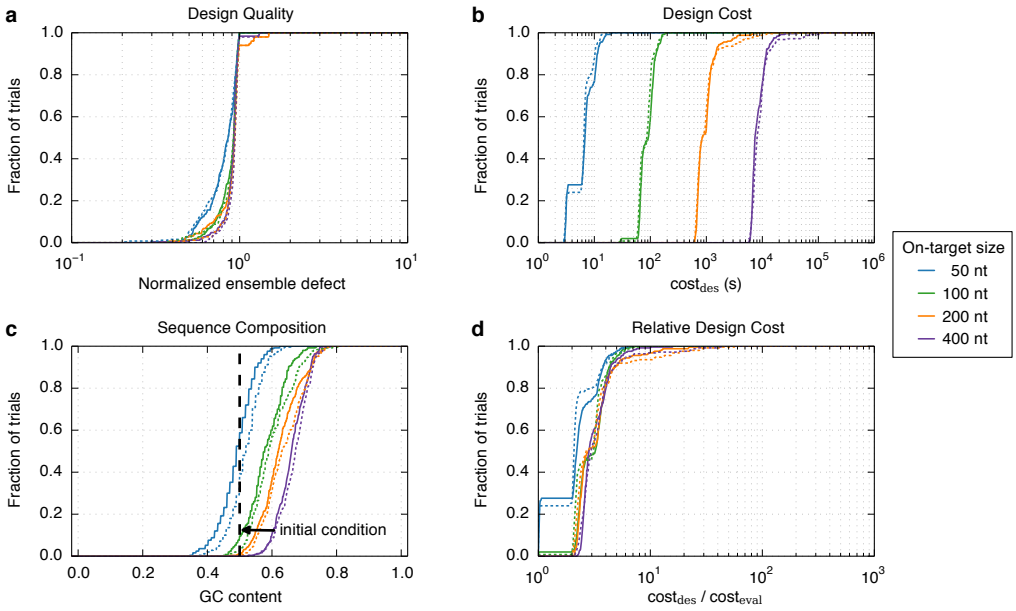


Figure 4.7: Multistate algorithm performance for test tube design. Algorithm performance for test tube design. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed line. d) Cost of sequence design relative to a single evaluation of the objective function. RNA design at 37 °C for the standard test set for the engineered test set (solid lines) and random test set (dashed lines).

For the special case of test tube design, the multiobjective design algorithm was compared to the test tube design algorithm for the design of multiple complexes in dilute solution. Figure 4.7 demonstrates that both algorithms behave similarly, as expected, since these algorithms are nearly identical for the special case of single test tube design.

4.4.2 Design of nucleic acid reaction pathways

One to five orthogonal instantiations of each system type were specified in each design. Ten trials were run for each of the resulting 25 designs. The stop condition, f^h_{stop}, was set to 0.05 for each objective. The median values for $\mathcal{M}$, the design cost, and design cost normalized by the analysis cost are shown in Figure 4.8. The average of the stop conditions was below unity, indicating that the stop conditions were achieved on average (panel a). Notably, the algorithm succeeds in generating five orthogonal designs for conditional dicer substrate production, even though library constraints imposed sequence identity with subsequences from the same two target mRNAs for each instantiation. The design costs increase as the cost of evaluating the multistate defect. The cost relative to the cost of evaluating the multistate defect sometimes remains the same, but sometimes increases. The logic gates and cooperative gates show nearly constant relative costs. Designs for HCR, conditional Dicer, and catalytic 3-arm junction formation all show an increase in relative design cost as the

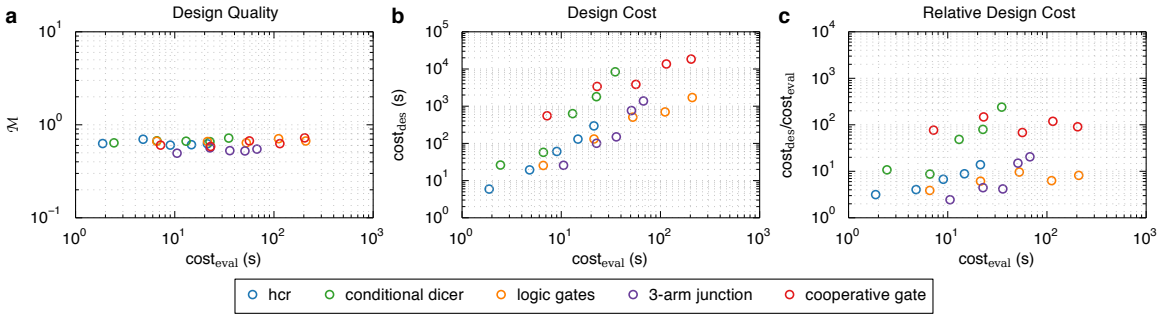


Figure 4.8: Multistate design performance. a) Multistate defect. b) Design cost. c) Normalized design cost shown vs the size of the analysis problem, characterized by the cost of evaluating the full multistate defect. Colors indicate the system type.

number of orthogonal instantiations is increased.

An example result for the design of two orthogonal HCR systems is depicted in Figure 4.9. The stop condition was achieved for every target test tube. Comparing this with the HCR objective specification in 4.1, we can see that the concentrations are similar to the target concentrations. Each tube is dominated by the on-target complexes and each structural ensemble is dominated by its target structure.

4.4.3 Preventing sequence patterns

Adding prevented patterns keeps particular subsequences from appearing on a specified strand or set of strands. For example, we can prevent any nucleotide from showing up four positions in a row by preventing the patterns `AAAA`, `CCCC`, `GGGG`, `UUUU`, or we could prevent any pair of nucleotides from appearing in six consecutive positions by preventing the patterns `RRRRRR`, `YYYYYY`, `MMMMMM`, `KKKKKK`, `WWWWWW`, `SSSSSS`. The design performances while preventing four consecutive instances of the same nucleotide or both four consecutive of the same nucleotide and six consecutive of any two nucleotides are shown in Figure 4.10, along with the original design performance. Preventing any repeats of a single nucleotide four times in a row does not have substantial effect on design quality or cost. Preventing a single nucleotide from appearing four consecutive times and any pair of nucleotides six consecutive times has an effect on two of the designs. The cost of design for conditional Dicer systems increases by nearly an order of magnitude when these constraints are used, and the cost of HCR systems increases by a factor of one to two when these constraints are included. Remarkably, the conditional Dicer designs nearly satisfy their stop conditions even under these additional constraints. The remaining design types do not show a large effect on design quality or cost when these constraints are included.

Example sequences are shown in Table 4.2 for a single HCR system. Long repeats observed

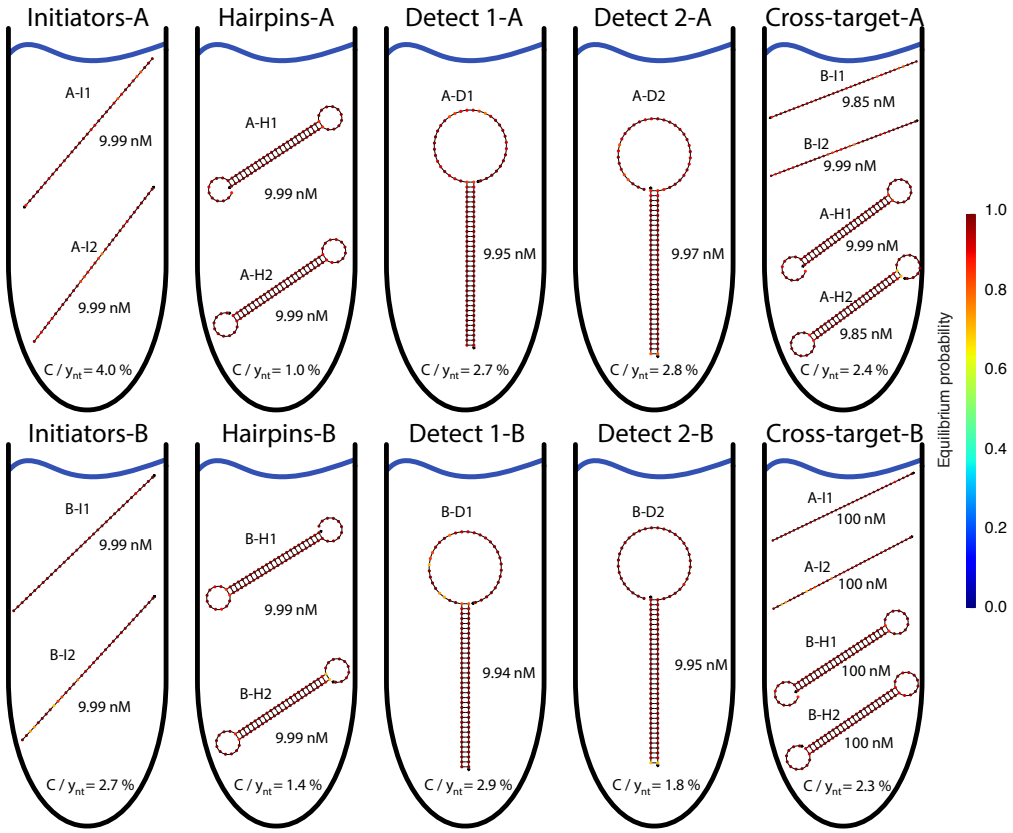


Figure 4.9: Summary of an HCR design of two systems intended to perform orthogonally. Each nucleotide is depicted in its intended state and shaded according to its probability of being in the depicted state. Compare to the specification in Figure 4.1. The first row of tubes corresponds to the specification for the first instantiation of HCR, and the second row of tubes corresponds to the second instantiation. The 'Cross-target' tubes minimize cross-talk between the two instantiations. The normalized test tube ensemble defect for each tube is shown at the bottom of the tube.

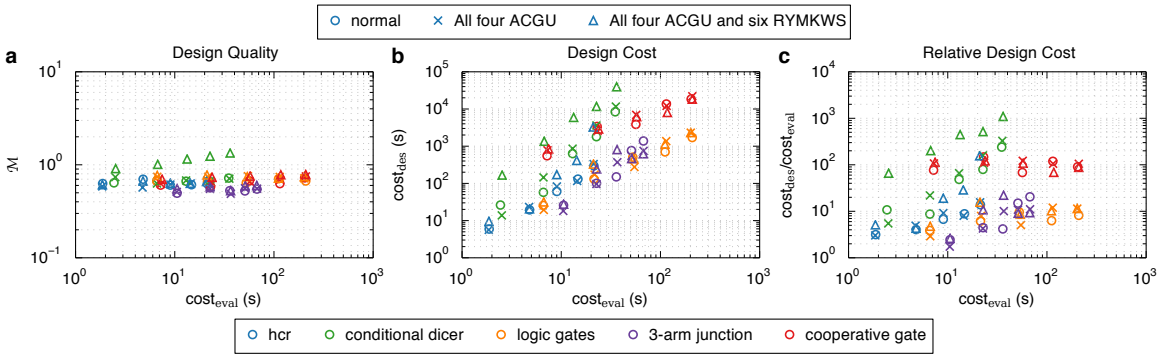


Figure 4.10: Effect of preventing sequence patterns. a) Design quality. b) Cost of design. c) Cost of design relative to cost of evaluation. Colors indicate the design type. Marker shapes indicate the type of constraint used.

without pattern preventions are eliminated when using explicit pattern prevention.

No prevented patterns

0-h1	GTAAGGGGGAAAGTAAGTAAGGAAAAAGCAAAAAACCTTCGATATTTTTGCTTTTTCCTTACTTACTT
0-h2	TTTTTGCTTTTTCCTTACTTACTTTCCCCCTTACAAGTAAGTAAGGAAAAAGCAAAAATATCGAAGGT
0-i1	TTTTTGCTTTTTCCTTACTTACTTTCCCCCTTAC
0-i2	ACCTTCGATATTTTTGCTTTTTCCTTACTTACTT

Prevent: AAAA, CCCC, GGGG, and UUUU

0-h1	TCAAACCATCACTCTCCATCACCCTTCTCTACCTGGAGGAGAAGAGGTAGAGAAGGGTGATGGAGAGT
0-h2	AGGTAGAGAAGGGTGATGGAGAGTGATGGTTTGAACTCTCCATCACCCTTCTCTACCTCTTCTCCTCC
0-i1	AGGTAGAGAAGGGTGATGGAGAGTGATGGTTTGA
0-i2	GGAGGAGAAGAGGTAGAGAAGGGTGATGGAGAGT

Prevent: AAAA, CCCC, GGGG, UUUU, RRRRRR, RRRRRR, RRRRRR, RRRRRR, RRRRRR, RRRRRR, RRRRRR

0-h1	GGAAACGTGAAATGTAAAGGCTGGGATGGAATGTTCACCTTTACACATTCCATCCCAGCCTTTACATT
0-h2	ACATTCCATCCCAGCCTTTACATTTCACGTTTCCAATGTAAAGGCTGGGATGGAATGTGTAAAGGTGA
0-i1	ACATTCCATCCCAGCCTTTACATTTCACGTTTCC
0-i2	TCACCTTTACACATTCCATCCCAGCCTTTACATT

Table 4.2: Sequences designed with and without prevented patterns.

4.4.4 Constraining content

The match constraint can be used to specify similarity to any pattern. In particular, it can be used to specify sequence content for domains or strands. We demonstrate this by creating a set of designs that are constrained to have a GC content between 40% and 60% for every domain. We created another set of designs where no nucleotide type could make up more than 35% of any domain longer than 3-nt.

The effect of these sequence content constraints on the quality and cost of design is shown in Figure 4.11. The defect and cost both are worse under these constraints. Under the GC content

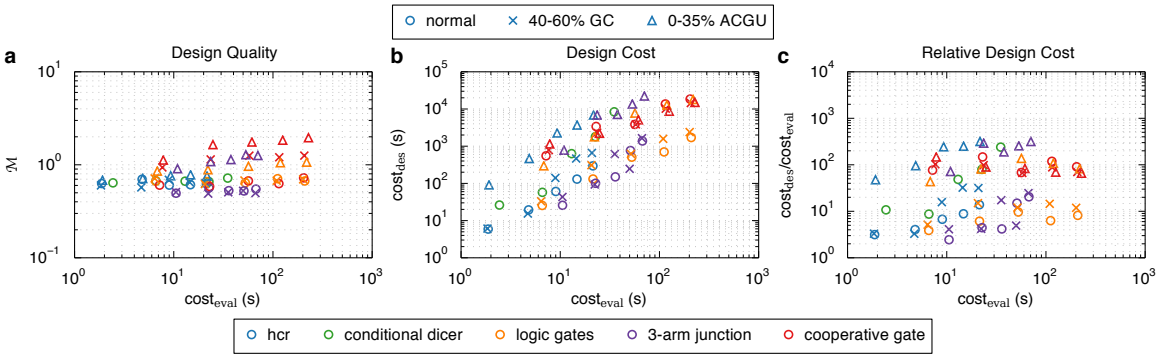


Figure 4.11: Effect of content constraints. a) Multistate defect. b) Cost of design. c) Cost of design relative to cost of evaluation. Colors indicate the design type. Marker shapes indicate the type of constraint used.

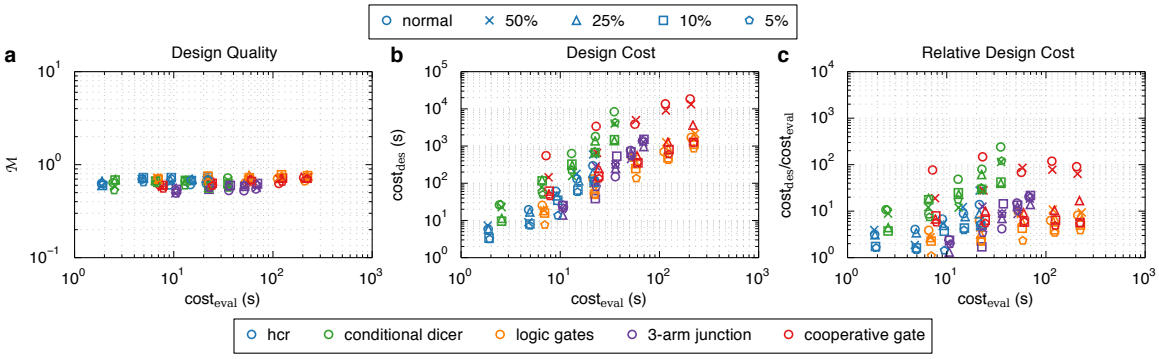


Figure 4.12: Effect of including structural defect weights. a) Multistate defect. b) Cost of design. c) Cost of design relative to cost of evaluation. Colors indicate the design type, marker shapes indicate the value of weight used. All weights were applied to all unstructured regions except for the toeholds used in subsequent reaction steps.

constraint, the cooperative gates typically fail to satisfy their stop condition. The ensemble defect shows deterioration of design quality when no nucleotide is allowed to constitute more than 35% of any domain. In several cases, the design cost increases by over an order of magnitude using this set of constraints. Note that the conditional Dicer designs cannot satisfy the constraints. Instead, these designs correctly warn that the constraints cannot be satisfied, and they permit no feasible sequences.

4.4.5 Weighting structural defects

Figure 4.12 shows the performance of the test tube design algorithm when using strucural defect weighting. All single-stranded regions except toeholds were assigned a reduced weight $\alpha^a_{h,j} \in \{0.05, 0.10, 0.25, 0.50\}$. Migration regions had their structural defect weight reduced from 1.00 to 0.50, 0.25, 0.10, and 0.05. Reducing the weight of the structural defects from unity makes the stop condition less stringent, and so typically results in lower design cost and satisfaction of more

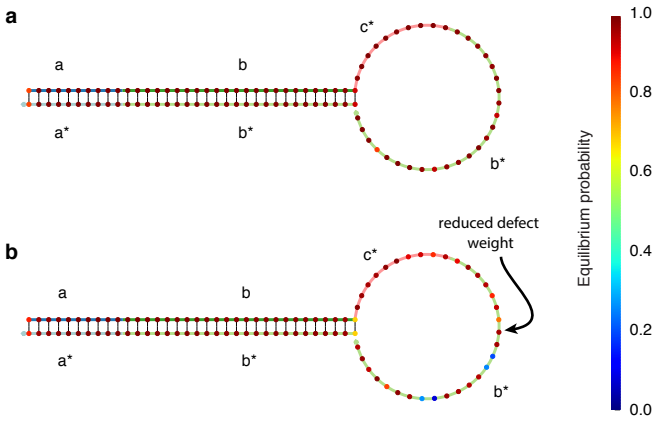


Figure 4.13: Detect 1 target structure from an HCR design designed without (panel a) and with (panel b) structural defect weighting to reduce the contribution to the structural defect contribution of the single-stranded domain b* (indicated). The toehold and stem remain at full weight.

stop conditions. The effect of using structural weights is shown in Figure 4.13 for the Detect 1 target structure from a designed HCR system. Panel a shows the target structure when the default weights of 1.0 are used everywhere. Panel b shows the target structure designed with nucleotide defect weights of 0.25 on the exposed branch migration region b^*. The complex designed with all weights equal to unity shows reduced base-pairing in the exposed b^* region, while the structure with reduced weights there maintains low nucleotide defects in the stem and in the toehold, but allows some pairing in the exposed b^* domain.

4.5 Conclusion

Multistate sequence design provides a powerful framework for optimizing the structures and binding energetics of pathways of interacting nucleic acid strands. The desired equilibrium properties for each test tube are specified as an arbitrary number of on-target complexes, each with a target secondary structure and target concentration, and an arbitrary number of off-target complexes, each with vanishing target concentration. Given a set of target test tubes, the multistate defect quantifies their average normalized test tube ensemble defect. Multistate defect optimization implements a positive design paradigm (stabilizes on-targets in each tube) and a negative design paradigm (destabilizes off-targets in each tube) at the structural and test tube levels. The algorithm allows restriction of the feasible sequence space through specification of a set of combinatorial sequence constraints that are satisfied throughout optimization. Using hierarchical ensemble decomposition and ensemble focusing, it is feasible to design sequences for nucleic acid reaction pathways of practical interest to the molecular programming and synthetic biology communities at cost comparable with the cost of evaluating the pathway's thermodynamic properties.

4.6 Appendix and archive content

Appendix Section A.2 explains the algorithm used to generate structures for the engineered test set. Appendix C contains additional studies on algorithm performance, a specification of the NUPACK design language, a discussion of the supplementary archive content, and a brief discussion about implementing new constraints and new objective functions.

The supplementary archive contains simple design examples that demonstrate constraints and Python scripts used to generate the examples used for this chapter.

Chapter 5

Simulation-based coarse-graining of nucleic acid energy landscapes

In the previous chapter, we described an algorithm to design the equilibrium base-pairing properties of a set of dilute solutions and used this algorithm to design sequences for nucleic acid reaction pathways. Following sequence design and prior to strand synthesis, it would be desirable to screen sequences by their kinetic behavior. Several recently developed tools have shown utility in understanding the kinetics of nucleic acid interactions [53, 72]. It is still inconvenient to summarize the results of these simulations, enumerating probable states and kinetic rates between them. Here, we describe two algorithms to estimate dominant states and high-level kinetics via simulation-based coarse-graining. Using the secondary structure kinetic model from Section 2.3.2, we formulate *small box coarse-graining* to generate low-dimensional approximations of secondary structure master equations for small numbers of distinguishable nucleic acid strands in solution. We extend this formulation to *large box coarse-graining* to estimate the mass action kinetics for a test tube of interacting nucleic acid strands. We validate both coarse-graining approaches against finer-grained simulations. The small box coarse-graining algorithm follows closely from the algorithm presented in Jon Othmer's thesis [51]. Large box coarse-graining is based on collaborations with Victor Beck and Justin Bois, previous researchers in the Pierce Lab. This thesis contributes a unified way of presenting these algorithms.

Both algorithms construct a set of species that is partitioned into an unexplored and an explored subset. Initial species are discovered using simulations starting from user-supplied initial conditions. The rest of each algorithm iteratively selects a species and explores its outgoing rates by using stochastic secondary structure simulations and by enforcing detailed balance among the explored species. Each unexplored species is selected for exploration based on its probability or its concentration in a coarse-grained simulation. The set of explored species is expanded until most of the probability or mass is contained in the explored species over the entire timescale of interest.

5.1 Small box coarse-graining

Small box coarse-graining attempts to determine a low-dimensional approximation of the master equation governing secondary structure transitions for a small box containing a few distinguishable strands. Given a set of strand sequences, ϕ, a corresponding ensemble of possible secondary structures, Ω^*, and a rate matrix, $R(\phi)$, that specifies transition rates between structures in Ω^*, we can describe the probability of being in secondary structure $\omega \in \Omega^*$ at time t by the master equation,

$$\frac{d\bar{p}(t)}{dt} = R(\phi)\bar{p}(t), \tag{5.1}$$

where $\bar{p}(t)$ is a vector of secondary structure probabilities for the small box.

The master equation is a linear ODE, and so can be solved analytically or numerically if $R(\phi)$ is small enough. Given an initial configuration, ω^0, a relaxation time, τ_{relax}, and a maximum transition time, $\tau_{\max}$, small box coarse-graining attempts to find dominant macrostates, Ω, consisting of sets of structures that locally equilibrate over timescales much less than τ_{relax}, and to find transitions between these macrostates that occur slower than τ_{relax}, but faster than $\tau_{\max}$.

One could imagine trying to find a nearly block diagonal form of the rate matrix, where eigenvalues corresponding to transitions between blocks, λ_{slow}, are much smaller than eigenvalues corresponding to transitions within each block, λ_{fast}. If all eigenvalues, λ, could be partitioned into either λ_{fast} or λ_{slow} for some choice of timescale, τ_{relax}, via

$$\lambda \in \lambda_{\text{fast}} \gg \frac{1}{\tau_{\text{relax}}} \gg \lambda \in \lambda_{\text{slow}}, \tag{5.2}$$

we could approximate the dynamics of the system over timescales slower than τ_{relax} by calculating the local equilibrium distribution within each block (based on the eigenvectors corresponding to the smallest eigenvalues), and estimating transition rates between the blocks using λ_{slow}. This estimate would hinge on the assumption that timescale separation existed between the two sets of eigenvalues, and on the correct choice of τ_{relax} to lie between these sets.

In nucleic acid systems, such timescale separation often exists. The fastest unimolecular transitions occur with rates on the order of 1×10^8/s, while the slowest transitions can occur at least four orders of magnitude slower [51, 64, 72]. Also, for most *in vitro* nucleic acid experiments, the rate of nucleic acid molecule collisions is typically much slower than the rate of intramolecular relaxation. For the purposes of this algorithm, we assume τ_{relax} exists and that the user has chosen it appropriately. We further require the user to choose a maximum simulation time, $\tau_{\max}$. Coarse-grained simulations and rate determination steps are run for at most $\tau_{\max}$ simulated time.

The rate matrix, $R(\phi)$, is typically much too large to enumerate. Direct simulation becomes infeasible for 25-nt sequences due to storage and computational constraints. Scaling to practical

systems requires efficient ways to represent and prune the state space [23, 41]. Instead, we use the Monte Carlo algorithm developed in Schaeffer's thesis, `multistrand`, to simulate secondary structure dynamics [64]. Each Monte Carlo simulation of the secondary structure master equation (5.1) results in a trajectory. We write the secondary structure at time t in trajectory i as ω_i^t. The interval of trajectory i in the range $[t_1, t_2]$ is written $\omega_i^{t_1:t_2}$.

In Section 5.1.1, we outline the algorithm, introducing key concepts. In Section 5.1.2, we describe the concepts in more detail.

5.1.1 Overview

Small box coarse-graining uses Monte Carlo trajectories to discover a set of macrostates based on a user-provided sequence, initial secondary structure, relaxation timescale, and maximum timescale. Discovering a set of macrostates, Ω, and transition rates between them, r. Each macrostate is a set of secondary structures that approximately reach local equilibrium over timescale τ_{relax}, as determined by comparing adjacent τ_{relax}-long windows. Macrostates are written Ω_h, where h is a unique index. In the matrix depiction, each macrostate corresponds to a block in the matrix. Transition rates r characterize transitions that occur on timescales longer than τ_{relax}, but shorter than $\tau_{\max}$. These rates correspond to eigenvalues in λ_{slow}. This section outlines the progression of the coarse-graining algorithm.

First, the algorithm discovers initial macrostates and estimates their starting probabilities. To do this, a trajectory is simulated, starting from the user-specified initial condition, until it reaches a local equilibrium, determined by comparing two adjacent windows of length τ_{relax}. This local equilibrium is a macrostate, Ω_h. It is added to the set of discovered macrostates, Ω, and a new trajectory is started from ω^0. Subsequent trajectories are run either until they are similar to a previously discovered macrostate, Ω_h, or until they reach local equilibrium in a new macrostate (which is saved as before). This process is repeated many times to discover the possible starting macrostates. The fraction of trajectories that end in each macrostate, Ω_h, is used to estimate its initial probability in coarse-grained master equation simulations, $p_h(0)$.

For the rest of this algorithm, the set of macrostates, Ω, is partitioned into an unexplored subset, Ω^{u}, and an explored subset, Ω^{x}. Immediately after discovery, macrostates are unexplored. Unexplored macrostates can only have incoming transitions. Explored macrostates can have both incoming and outgoing transitions, and the partition function for each explored macrostate is estimated. Using this partition function estimate, detailed balance is enforced between macrostates in Ω^{x}. After discovering initial macrostates (which are added to Ω^{u}) and estimating their initial probabilities, the rest of the algorithm consists of iteratively selecting a state from Ω^{u}, transferring it to Ω^{x}, estimating its outgoing rates, and estimating its partition function. This process is depicted in Figure 5.1. Each panel will be referenced in the appropriate paragraph of this overview. The

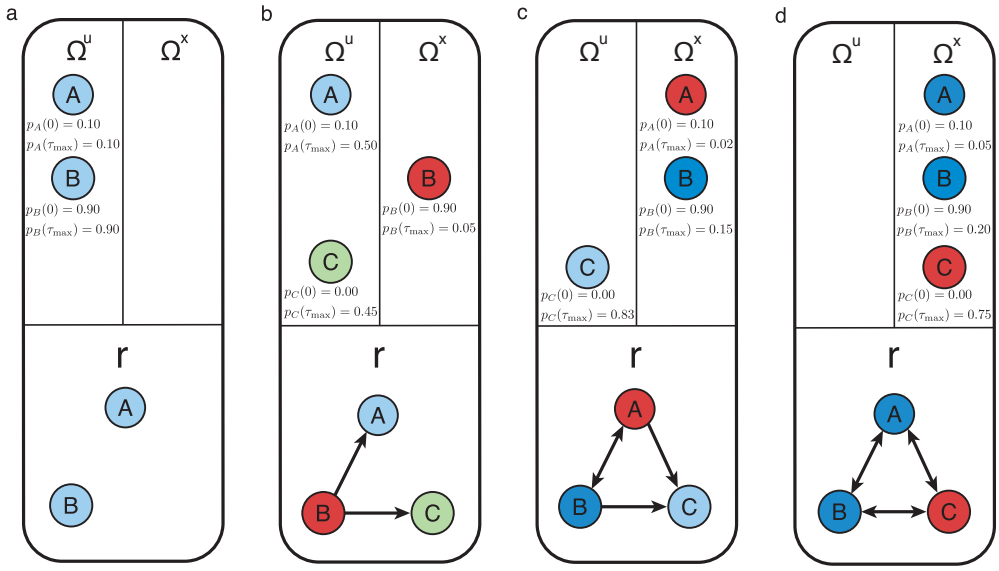


Figure 5.1: The algorithm is initialized with a set of sequences and an initial secondary structure. Trajectories are simulated starting from this secondary structure, and all discovered basins are added to Ω^{u} (panel a). No rate information is available so $p_h(\tau_{\max}) = p_h(0)$. Next, the macrostate with the highest probability, B, is added to Ω^{x}, and simulations are run from the MFE of this basin until $\tau_{\max}$ total time has been simulated in that macrostate. Newly discovered macrostate, C, is added to Ω^{u} (panel b). The remaining macrostates are simulated and added to the set of reactions until the total probability in Ω^{u} at $\tau_{\max}$ is less than p_{stop} (panels c and d). In panel d, all macrostates are in Ω^{x}, and the algorithm terminates.

state of the algorithm before exploration commences is shown in panel a: macrostates A and B were discovered during the initial stage of the algorithm, added to Ω^{u}, and their initial probabilities were estimated.

After estimating the initial probabilities, the macrostate, $\Omega_{\hat{h}}$, with the highest estimated starting probability, $p_{\hat{h}}(0)$, is transferred to Ω^{x} and outgoing transition rates are estimated using *macrostate exploration*. Outgoing rates are initially estimated by simulating long trajectories that start in the selected macrostate until they enter a different macrostate. The entered macrostate can be a known macrostate or a newly discovered macrostate, discovered when the trajectory reaches local equilibrium. After simulating $\tau_{\max}$ total time starting from the selected macrostate, the outgoing transition rate to each other macrostate, $r_{\hat{h}\to j}, j \neq \hat{h}$, is estimated based on the number of observed transitions to it. The previously estimated initial macrostate probabilities and newly estimated macrostate transition rates define a coarse-grained Markov Chain that is now simulated. Figure 5.1b represents this stage in the algorithm. Macrostate B was explored since it had the highest initial probability. Transition rates to previously discovered macrostate A and newly discovered macrostate C were estimated by simulating long trajectories starting in B. The final probabilities, $p_h(\tau_{\max})$, were updated by simulating the coarse-grained master equation.

After the coarse-grained simulation, the unexplored macrostate, $\Omega_h \in \Omega^{\mathrm{u}}$, with the highest

probability at the end of the simulation is transferred to Ω^{x}, and its outgoing rates are estimated in a subsequent round of macrostate exploration. After performing the trajectory simulations and calculating transition rates as before, detailed balance is enforced between all macrostates in Ω^{x} using estimates of the macrostate partition functions. This stage of the algorithm is shown in Figure 5.1c. Macrostate A was chosen for the second round of exploration. The rates between A and B satisfy detailed balance, but transitions to macrostate C are unidirectional since it is not yet explored.

The total probability in Ω^{u} at time τ_{max} is a quantification of the error in the coarse-graining that can be eliminated by further rounds of exploration. Macrostate exploration is repeated, selecting the macrostate with the highest probability at time τ_{max} each round until this coarse-graining error is less than a user-specified stop probability, p_{stop}. Figure 5.1d shows the state of the algorithm after the third round of exploration. All macrostates are now explored, so detailed balance is enforced for the entire coarse-grained Markov chain and the coarse-graining algorithm terminates.

5.1.2 Algorithm

There are three important ingredients for this algorithm: characterization of macrostates, initial state discovery based on short trajectories starting from the initial secondary structure, and transition discovery based on long trajectory simulations starting within each macrostate. In this section, we explain each of these: Section 5.1.2.1 describes how distance metrics are used to discover new macrostates and match known macrostates, Section 5.1.2.2 describes how starting macrostates are identified and how their starting probabilities are estimated, and Section 5.1.2.3 describes the process of selecting a macrostate, estimating its outgoing transition rates, and simulating the coarse-grained master equation.

5.1.2.1 Characterizing trajectories

To identify macrostates using trajectories, we compare trajectory windows of length τ_{relax}, quantifying their similarity. This similarity metric will be used to determine when a trajectory is in local equilibrium, i.e., when a trajectory is approximately ergodic. The metric will also be used to determine when a trajectory is in a known macrostate. In this section, we first describe two distance metrics, the distance in variation and the base pair distance. Next, we introduce representative trajectories and representative structures for macrostates. Finally, we describe how we use the distance metrics and representative trajectories to discover new macrostates and to determine when a trajectory is in a known macrostate.

Distance metrics. The distance in variation, d^V, between two trajectory regions $\omega_i^{0:\tau_{\text{relax}}}$ and $\omega_j^{0,\tau_{\text{relax}}}$ is half the L_1 distance between the sampled secondary structure probability vectors [28, 51]:

$$d^V(\omega_i, \omega_j) = \frac{1}{2} \sum_{\omega \in \Omega^*} \left| \frac{\int_0^{\tau_{\text{relax}}} \delta(\omega, \omega_i^t) dt}{\tau_{\text{relax}}} - \frac{\int_0^{\tau_{\text{relax}}} \delta(\omega, \omega_j^t) dt}{\tau_{\text{relax}}} \right|, \tag{5.3}$$

where $\delta(\omega, \omega_i^t)$ is equal to unity if $\omega = \omega_i^t$ and is equal to zero otherwise. The total distance is normalized by 1/2 to take a value between zero and unity. The distance in variation is zero if and only if the sampled distributions are exactly equal. The distance is unity if there is no overlap between the two sets of sampled secondary structures. This distance metric was used to determine the location of transitions between macrostates in the initial macrostate discovery stage of Othmer's algorithm [51]. This distance metric has the advantage that it relies solely on the fraction of time spent in each state and makes no assumptions about the underlying Markov chain. It has the disadvantage that, for larger nucleic acid systems, the number of states that are nearly isoenergetic increases rapidly, and it takes a correspondingly long time to approach ergodicity among individual states.

To find an alternative metric, we note that the distance in variation treats each pair of structures as if they were completely different, ignoring structural similarities. Also, in Othmer's work, each macrostate is characterized by its pair probability matrix during macrostate clustering; macrostates are clustered based on the L_1 norm between their sampled pair probability matrices. This metric is convenient because the space used by pair probability matrices scales quadratically with the number of nucleotides and the base pair probabilities converge more rapidly than secondary structure probabilities (see Figure 5.3ab). To take advantage of this faster convergence and reduced storage use, we define the trajectory base pair distance as the L_1 norm between the sampled pair probability matrices normalized to lie between zero and unity. For trajectory ω_i starting at t_0 and ending at t_f, we define the sampled pair probability matrix $\tilde{P}(\omega_i^{t_0:t_f})$ as

$$\tilde{P}^{a,b}(\omega_i^{t_0:t_f}) \equiv \frac{1}{t_f - t_0} \int_{t_0}^{t_f} S^{a,b}(\omega_i^t) dt, \tag{5.4}$$

where $S^{i,j}$ is the natural extension of the structure matrix to the small box, as defined in Chapter 2. The base pair distance between two trajectories, ω_i and ω_j, is then

$$d^P(\omega_i, \omega_j) = \frac{1}{2|\phi|} \sum_{\substack{1 \le a \le |\phi_j| \\ 1 \le b \le |\phi_j|+1}} \left| \tilde{P}^{a,b}(\omega_i) - \tilde{P}^{a,b}(\omega_j) \right|. \tag{5.5}$$

This is normalized by twice the number of nucleotides, so it falls between zero and unity. Note that it can be zero even if the sampled distributions are unequal. Therefore, it is not a distance metric over the space of possible probability distributions, only over the sampled pair probability matrices.

The lost information is the difference in correlations between base pairs. Within a macrostate, we typically don't care about these differences in correlation, so we use the base pair distance to take advantage of its faster convergence and lower storage use when compared to the distance in variation. A computationally expensive alternative that captures the correlation in base pairs is the Wasserstein distance. A good survey of probability metrics and how they relate is given by Gibbs [28].

Representing macrostates. To use the distance metrics, we associate each macrostate, Ω_h, with a representative trajectory, $\omega_{\Omega_h}^{0:\tau_{\text{relax}}}$. This is used to identify entries to and exits from the macrostate during subsequent trajectory simulations. For practical purposes, the full trajectory need not be saved; if the base pair distance is used, only the sampled pair probability matrix needs to be saved.

When trajectories need to be started from within a macrostate, we select the initial secondary structure based on the representative trajectory. Othmer argued that the centroid secondary structure, which is the structure with the minimal ensemble defect according to the sampled pair probability matrix,

$$\omega_{\Omega_h}^{\text{centroid}} = \underset{\omega \in \Omega^*}{\text{argmin}} \left[|\phi| - \sum_{\substack{1 \le a \le |\phi| \\ 1 \le b \le |\phi|+1}} \tilde{P}^{a,b}(\omega_{\Omega_h}) S^{a,b}(\omega) \right] \tag{5.6}$$

should be used for this purpose. This has the disadvantage of possibly choosing a structure that has an arbitrarily low probability compared to the structures in the representative trajectory (see Figure 5.2). In this work, we choose to use the sampled minimum free energy structure as the starting structure for trajectories instead:

$$\omega_{\Omega_h}^{\text{MFE}} \equiv \underset{\omega_{\Omega_h}^t \in \omega_{\Omega_h}^{0:\tau_{\text{relax}}}}{\text{argmin}} \left[\Delta G \left(\omega_{\Omega_h}^t \right) \right]. \tag{5.7}$$

This has the highest probability of any sampled structure in the trajectory.

Trajectory distance criteria. At any point in the algorithm, we determine that a trajectory has locally equilibrated when the previous and next windows of length τ_{relax} are approximately the same, indicating the trajectory is approaching ergodicity on this timescale. We characterize this using the *local equilibration criterion*,

$$d^P(\omega_i^{t-\tau_{\text{relax}}:t}, \omega_i^{t:t+\tau_{\text{relax}}}) < f_{\text{equil}}, \tag{5.8}$$

where $f_{\text{equil}} \in (0, 1)$. This criterion is used to identify new macrostates. The parameter f_{equil} specifies the stringency of the equilibration criterion.

In addition to the local equilibration criterion, we wish to be able to determine if a trajectory is

in a known macrostate. To do this, we measure the difference between the representative trajectory, $\omega_{\Omega_h}^{0:\tau_{\text{relax}}}$, for each macrostate, Ω_h, and the next window of length τ_{relax}. A trajectory window, $\omega_i^{t:t+\tau_{\text{relax}}}$, *matches* Ω_h if it satisfies the *macrostate match criterion*:

$$d^P(\omega_i^{t:t+\tau_{\text{relax}}}, \omega_{\Omega_h}^{0:\tau_{\text{relax}}}) < f_{\text{match}}. \tag{5.9}$$

The macrostate match criterion is used to determine entry and exits times for known macrostates. We typically set $f_{\text{match}} > f_{\text{equil}}$ so known macrostates are preferentially matched using (5.9), and rediscovery of the same macrostate using (5.8) is avoided.

5.1.2.2 Identifying initial macrostates

Given an initial secondary structure, ω^0, the first step of the algorithm identifies the macrostates it relaxes into. A set of M_{init} trajectories starting from ω^0 are run. The first trajectory is run until the local equilibration criterion (5.8) is satisfied and the first macrostate Ω_h is initialized and added to the set of unexplored macrostates, Ω^{u}. The corresponding representative trajectory is assigned to the window of length τ_{relax} centered at equilibration time t,

$$\omega_{\Omega_h}^{0:\tau_{\text{relax}}} \equiv \omega_i^{t-\frac{\tau_{\text{relax}}}{2}:t+\frac{\tau_{\text{relax}}}{2}}, \tag{5.10}$$

and the representative MFE structure is assigned using (5.7). The initial entry counter, m_{init}^h, keeps track of the number of transitions into macrostate Ω_h from the initial secondary structure. Upon macrostate identification, this counter is initialized to unity.

Subsequent trajectories are run until they either satisfy the local equilibration condition (5.8) (in which case the new macrostate, Ω_h, is saved and the corresponding counter m_{init}^h is initialized to unity), or until they satisfy the macrostate match criterion (5.9) with any previously identified macrostate, Ω_h, (in which case the corresponding counter, m_{init}^h, is incremented). After simulating M_{init} trajectories, the probability of starting in each macrostate Ω_h is estimated as

$$p_h(0) = \frac{m_{\text{init}}^h}{M_{\text{init}}}. \tag{5.11}$$

These probabilities constitute the initial conditions for the coarse-grained master equation simulations.

5.1.2.3 Macrostate exploration

After identifying initial macrostates and their initial probabilities, the algorithm iteratively explores macrostates, estimating their outgoing transition rates and transferring them from Ω^{u} to Ω^{x}. Outgoing transition rates are estimated by simulating long trajectories and by enforcing detailed balance

among explored macrostates. Macrostates are selected for exploration based on their probability during coarse-grained simulations.

Simulating outgoing transitions. During the first iteration of macrostate exploration, the macrostate, $\Omega_{\hat{h}}$, with the highest initial probability, $p_{\hat{h}}(0)$, is chosen, and transitions away from this macrostate are explored. A set of trajectories is simulated, starting from the representative minimum free energy structure, $\omega_{\hat{h}}^{\mathrm{MFE}}$, for the chosen macrostate. Each trajectory can either a) leave the initial macrostate and enter an already discovered macrostate, b) leave the initial macrostate and reach local equilibrium in a newly discovered macrostate, or c) remain in the initial macrostate until $\tau_{\max}$ cumulative time has been simulated in that macrostate.

To determine if a trajectory has transitioned to a previously discovered macrostate, the distances between the next window of length τ_{relax} and the representative trajectories of previously discovered macrostates are constantly updated as the trajectory is simulated. At any time, t, if the trajectory transitions to a known macrostate, Ω_j, by satisfying the macrostate match criterion (5.9) with any macrostate except the initial macrostate, the trajectory is ended. A counter, $m_{\mathrm{trans}}^{\hat{h}\to j}$, keeps track of the number of observed transitions from the initial macrostate, $\Omega_{\hat{h}}$, to each destination macrostate, Ω_j.

To discover transitions to previously undiscovered macrostates, the distance between the previous window of length τ_{relax} and the next window of length τ_{relax} is constantly updated as the trajectory is simulated. If, at any point, the local equilibration criterion (5.8) is satisfied between these two windows, the trajectory is in a macrostate. If no known macrostates satisfy the macrostate match criterion with the next τ_{relax} window, then a new macrostate has been discovered. This new macrostate, Ω_j, its representative trajectory, $\omega_{\Omega_j}^{0:\tau_{\mathrm{relax}}}$, and its MFE structure, $\omega_{\Omega_j}^{\mathrm{MFE}}$, are saved as in initial macrostate discovery and the counter for that transition, $m_{\mathrm{trans}}^{\hat{h}\to j}$, is initialized to unity.

After $\tau_{\max}$ cumulative time has been simulated starting from the chosen macrostate, $\Omega_{\hat{h}}$, outgoing transition rates to all other macrostates, Ω_j, are estimated:

$$r_{\hat{h}\to j} \equiv \frac{m_{\mathrm{trans}}^{\hat{h}\to j}}{\tau_{\max}}. \tag{5.12}$$

Estimating partition functions. All transition rates between explored macrostates are required to satisfy detailed balance. For this, we need an estimate of the partition function of each macrostate. This is done by calculating the trajectory partition function, $\tilde{Q}_{\hat{h}}$, over the representative trajectory for the macrostate, $\omega_{\Omega_{\hat{h}}}$, and inflating it based on the estimated fraction of the macrostate it captures.

The trajectory partition function is the sum of Boltzmann weights over all structures in the

representative trajectory:

$$\tilde{Q}_{\hat{h}} \equiv \sum_{\omega^j_{\Omega_h} \in \omega^{0:\tau_{\text{relax}}}_{\Omega_h}} \exp(-\Delta G(\omega^j_{\Omega_h})/h_B T). \tag{5.13}$$

To estimate the fraction of the macrostate included in this sum, M_{pfunc} trajectories are then simulated for τ_{relax} time, starting from the minimum free energy structure, $\omega^{\text{MFE}}_{\Omega_h}$. The ending state, $\omega_i^{\tau_{\text{relax}}}$, for each trajectory is assumed to be a randomly sampled secondary structure from the macrostate. A counter, m_{pfunc}, keeps track of the number of times the sampled ending state is present in the representative trajectory. The probability of sampling a structure from the representative trajectory is given by

$$p(\omega_i^{\tau_{\text{relax}}} \in \omega^{0:\tau_{\text{relax}}}_{\Omega_h}) \approx \frac{\tilde{Q}_h}{Q_h}. \tag{5.14}$$

where Q_h is the true partition function of the macrostate. We then estimate the true partition function by the ratio

$$Q_h \equiv \frac{\tilde{Q}_h M_{\text{pfunc}}}{m_{\text{pfunc}}}. \tag{5.15}$$

For a discussion of the error bounds on rate estimates and partition functions, see Othmer's thesis [51].

Enforcing detailed balance. At this point, each macrostate in Ω^{x} has an estimate of its partition function and an estimate of its outgoing rates. When multiple states have been added to Ω^{x}, the partition functions can be used to enforce detailed balance between the explored macrostates. Typically, the energetically favorable direction for any transition will be simulated much more often than the unfavorable direction, so the estimate of the favorable transition rate will be more accurate. The coarse-graining algorithm keeps the rate for whichever transition was sampled more often and calculates reverse rates using detailed balance, e.g., for two states h and j, if $m^{h\to j}_{\text{trans}} > m^{j\to h}_{\text{trans}}$, we override any transition rate from Ω_j to Ω_h calculated with (5.12), with the rate calculated using detailed balance:

$$r_{j\to h} = r_{h\to j} \frac{Q_h}{Q_j}. \tag{5.16}$$

Macrostate selection and termination conditions. At this point in the algorithm, the macrostates in Ω^{x} all have estimates of their partition functions and estimates of their outgoing rates. Additionally, all transition rates between macrostates in Ω^{x} satisfy detailed balance. These transition rates and the previously determined initial conditions define a coarse-grained Markov chain. The master equation for this coarse-grained Markov chain is now simulated [32].

Note that transitions to states in Ω^{u} are irreversible since no outgoing rates are specified until after exploration. Thus, if $\Omega^{\text{u}} \neq \emptyset$, the coarse-grained Markov chain is decomposable and the

probability of being in a macrostate contained in Ω^{u} is non-decreasing as the coarse-grained master equation simulation proceeds. However, some macrostates in Ω^{u} may accumulate a negligible amount of probability after $\tau_{\max}$ time, and so have only a minor effect on the solution to the master equation. To enumerate macrostates with the highest probability first and to enable a user to stop the coarse-graining before all macrostates are explored, we define the unexplored probability as the total probability in all unexplored states at time $\tau_{\max}$:

$$p_{\Omega^{\mathrm{u}}} \equiv \sum_{\Omega_h \in \Omega^{\mathrm{u}}} p_h(\tau_{\max}). \tag{5.17}$$

Using this, we define the small box stop condition:

$$p_{\Omega^{\mathrm{u}}} \leq p_{\mathrm{stop}}, \tag{5.18}$$

where $p_{\mathrm{stop}} \in [0, 1)$ is a user-specified cutoff probability. The algorithm terminates when this is satisfied, which is guaranteed to occur when $\Omega^{\mathrm{u}} = \emptyset$, in which case $\Omega^{\mathrm{x}} = \Omega$.

If the stop condition is not satisfied, the unexplored macrostate with the highest probability at time $\tau_{\max}$,

$$\Omega_{\hat{h}} = \underset{\Omega_h \in \Omega^{\mathrm{u}}}{\mathrm{argmax}}[p_h(\tau_{\max})], \tag{5.19}$$

is selected and the next round of macrostate exploration commences, determining outgoing rates from this macrostate. This process is repeated until the stop condition is satisfied.

Looking back, we picked the macrostate with the highest initial probability for the first iteration of macrostate exploration. No transition rates were enumerated at this point; this was equivalent to performing a coarse-grained simulation of a constant master equation and choosing the macrostate based on its probability at $\tau_{\max}$ instead. We thus define the coarse-grained Markov chain after initial macrostate discovery to have no transitions and we simulate this master equation. This allows us to always choose the macrostate with the highest probability after $\tau_{\max}$ time according to the current coarse-grained master equation.

The pseudocode for small box coarse-graining is shown in Algorithm 5.1.

5.1.3 Methods

The small box coarse-graining algorithm was tested on a set of small single strands that exhibit multi-state behavior. For a particularly small strand, the resulting master equation solution was compared to the full, exhaustively enumerated, master equation solution.

The algorithm was also tested on an HCR system designed in section 4.4.2. This test is similar to the intended use: coarse-graining reaction pathways.

All single-stranded tests were run using RNA at 25 °C with no dangles. The HCR tests were

SmallboxCG(ω^0, ϕ)
  $\Omega^{\mathrm{u}}, \omega_\Omega, p_\Omega(0) \leftarrow$ FindInitialStates($\omega(0), \phi$)
  $p_h(\tau_{\max}) \leftarrow p_h(0)\ \forall \Omega_h \in \Omega$
  while $\sum_{j\in\Omega^{\mathrm{u}}} p_j(\tau_{\max}) > p_{\mathrm{stop}}$
    $\Omega\hat{h} \leftarrow \mathrm{argmax}_{\Omega_h\in\Omega^{\mathrm{u}}}\, p_h(\tau_{\max})$
    $\Omega^{\mathrm{x}} \leftarrow \Omega^{\mathrm{x}} \cup \{\Omega\hat{h}\}$
    $\Omega^{\mathrm{u}} \leftarrow \Omega^{\mathrm{u}} \setminus \{\Omega\hat{h}\}$
    $m_{\hat{h}\to\Omega}, \omega_\Omega, \Omega^{\mathrm{x}} \leftarrow$ ExploreState($\hat{h}, \omega_\Omega, \Omega, \phi$)
    $Q_{\hat{h}} \leftarrow$ EstimatePfunc($\omega_{\hat{h}}$)
    $r_{\Omega\to\Omega} \leftarrow$ EstimateRates($m_{\Omega\to\Omega}, Q_\Omega$)
    $p_\Omega(\tau_{\max}) \leftarrow$ SimulateCG($r_\Omega, p_\Omega(0), \tau_{\max}$)
  return $p_\Omega(0), Q_\Omega, r_{\Omega\to\Omega}$

FindInitialStates($\omega(0), \phi$)
  $\Omega^{\mathrm{u}}, \omega_\Omega \leftarrow \emptyset, \emptyset$
  for $i \in 1, \ldots, M_{\mathrm{init}}$
    $\Omega^{\mathrm{u}}, \omega_\Omega, h \leftarrow$ MatchTrajectory($\omega^0, \phi, \Omega^{\mathrm{u}}, \omega_\Omega$)
    $m^h_{\mathrm{init}} \leftarrow m^h_{\mathrm{init}} + 1$
  $p_h(0) = m^h_{\mathrm{init}}/M_{\mathrm{init}}\ \forall \Omega_h \in \Omega^{\mathrm{u}}$
  return $\Omega^{\mathrm{u}}, \omega_{\Omega^{\mathrm{u}}}, p_{\Omega^{\mathrm{u}}}$

MatchTrajectory($\omega^0, \phi, \Omega, \omega_\Omega$)
  $t \leftarrow -\tau_{\mathrm{relax}}$
  $\hat{\omega} \leftarrow \omega^0$
  while $t < t^f$ **and** $d^P(\omega^{t-\tau:t}, \omega^{t:t+\tau_{\mathrm{relax}}}) \geq f_{\mathrm{equil}}$
      $d^P(\omega_h, \omega^{t,t+\tau}) \geq f_{\mathrm{match}} \forall h \in \Omega$
    $\omega', t' \leftarrow$ MCStep($\hat{\omega}$)
    $\omega^{t+\tau_{\mathrm{relax}}:t+\tau_{\mathrm{relax}}+t'} \leftarrow \hat{\omega}$
    $t \leftarrow t + t'$
    $\hat{\omega} \leftarrow \omega'$
  if $\exists \Omega_h \in \Omega : d^P(\omega_{\Omega_h}, \omega^{t,t+\tau_{\mathrm{relax}}}) < f_{\mathrm{match}}$
    $\hat{h} \leftarrow \mathrm{argmin}_{\Omega_h\in\Omega}(d^P(\omega_{\Omega_h}, \omega^{t,t+\tau_{\mathrm{relax}}}))$
  else
    $\hat{h} \leftarrow |\Omega| + 1$
    $\Omega \leftarrow \Omega \cup \{\hat{h}\}$
    $\omega_{\Omega_{\hat{h}}} \leftarrow \omega^{t-\frac{\tau_{\mathrm{relax}}}{2}, t+\frac{\tau_{\mathrm{relax}}}{2}}$
  return $\hat{h}, \Omega, \omega_\Omega$

ShortTrajectory(ω^0, ϕ, t^f)
  $c^{\mathrm{match}} \leftarrow$ **true**
  $\hat{\omega} \leftarrow \omega^0$
  $t \leftarrow 0$
  while $t < t^f$
    $\omega', t' \leftarrow$ MCStep($\hat{\omega}$)
    $\omega^{t:t+t'} \leftarrow \hat{\omega}$
    $t \leftarrow t + t'$
    $\hat{\omega} \leftarrow \omega'$
  return $\omega^{0:t^f}$

ExploreState($\hat{h}, \omega_\Omega, \Omega, \phi$)
  $t \leftarrow 0$
  $m_{\hat{h}\to j} \leftarrow 0\ \forall \Omega_j \in \Omega$
  while $t < \tau_{\max}$
    $j, t', \Omega, \omega_\Omega \leftarrow$
      LongTrajectory($\omega^{\mathrm{MFE}}_{\Omega_{\hat{h}}}, \phi, \Omega, \omega_\Omega, t - \tau_{\max}$)
    $t \leftarrow t + t'$
    $m_{\hat{h}\to j} \leftarrow m_{\hat{h}\to j} + 1$
  return $m_{\hat{h}\to\Omega}, \omega_\Omega, \Omega$

LongTrajectory($\omega^0, \phi, \Omega, \omega_\Omega, \tau^f, \hat{h}$)
  $t \leftarrow -\tau_{\mathrm{relax}}$
  $\hat{\omega} \leftarrow \omega^0$
  while $t < t^f$ **and** $d^P(\omega_h, \omega^{t,t+\tau}) \geq f_{\mathrm{match}} \forall h \in \Omega, h \neq \hat{h}$
      $d^P(\omega^{t-\tau:t}, \omega^{t:t+\tau_{\mathrm{relax}}}) \geq f_{\mathrm{equil}}$
    $\omega', t' \leftarrow$ MCStep($\hat{\omega}$)
    $\omega^{t+\tau_{\mathrm{relax}}:t+\tau_{\mathrm{relax}}+t'} \leftarrow \hat{\omega}$
    $t \leftarrow t + t'$
    $\hat{\omega} \leftarrow \omega'$
  if $\exists \Omega_h \in \Omega : d^P(\omega_{\Omega_h}, \omega^{t,t+\tau_{\mathrm{relax}}}) < f_{\mathrm{match}}$
    $j \leftarrow \mathrm{argmin}_{\Omega_h\in\Omega}(d^P(\omega_{\Omega_h}, \omega^{t,t+\tau_{\mathrm{relax}}}))$
  else
    $j \leftarrow |\Omega| + 1$
    $\Omega^{\mathrm{u}} \leftarrow \Omega^{\mathrm{u}} \cup \{j\}$
    $\omega_{\Omega_j} \leftarrow \omega^{t-\frac{\tau_{\mathrm{relax}}}{2}, t+\frac{\tau_{\mathrm{relax}}}{2}}$
  return $j, t, \Omega, \omega_\Omega$

EstimatePfunc($\omega_{\Omega_{\hat{h}}}$)
  $\tilde{Q}_h \leftarrow \sum_{\omega\in\omega_{\Omega\hat{h}}} \exp\left[-\frac{\Delta G(\omega)}{h_B T}\right]$
  for $i \in 1, \ldots, M_{\mathrm{pfunc}}$
    $\omega_i \leftarrow$ ShortTrajectory($\omega^{\mathrm{MFE}}_{\Omega\hat{h}}, \phi, \tau_{\mathrm{relax}}$)
    if $\omega_i^{\tau_{\mathrm{relax}}} \in \omega_{\Omega\hat{h}}$
      $m_{\mathrm{pfunc}} \leftarrow m_{\mathrm{pfunc}} + 1$
  $Q_h \leftarrow \frac{\tilde{Q}_h M_{\mathrm{pfunc}}}{m_{\mathrm{pfunc}}}$
  return Q_h

EstimateRates($m_{\Omega\to\Omega}, Q_\Omega$)
  $r_{\Omega\to\Omega} \leftarrow 0, \ldots, 0$
  for $\Omega j_1, \Omega_{j_2} \in \Omega^{\mathrm{x}} \times \Omega$
    if $m_{j_2\to j_1} > m_{j_1\to j_2}$
      $j_1, j_2 = j_2, j_1$
    $r_{j_1\to j_2} \leftarrow m_{j_1\to j_2}/\tau_{\max}$
    if $\Omega_{j_2} \in \Omega^{\mathrm{x}}$
      $r_{j_2\to j_1} \leftarrow r_{j_1\to j_2} Q_{j_1}/Q_{j_2}$
  return $r_{\Omega\to\Omega}$

Algorithm 5.1: Small box coarse-graining. Given a sequence ϕ and initial secondary structure ω^0, this algorithm identifies the dominant macrostates and transition rates for the small box kinetic problem. Each Monte Carlo step is picked using the MCStep process. To ensure the next τ_{relax} window is defined for positive times t, we start each simulation at $t = -\tau_{\mathrm{relax}}$ and append each Monte Carlo step at $t + \tau_{\mathrm{relax}}$ in both MatchSimulation and LongSimulation. We use the convention that the distance d^P to an incomplete window is infinite.

run using DNA at 25 °C with no dangles.

5.1.3.1 Implementation

This algorithm was coded in C, calling the `multistrand` C++ library to implement the secondary structure kinetic model [64] and ODEPACK Fortran library to simulate the coarse-grained master equation [32]. The exhaustively enumerated solution was coded in Python using the SciPy library [37], and all graphs were generated in Python using `matplotlib` [35].

5.1.4 Results

The main results demonstrate the effectiveness of the choices for the representative structure and distance metric. We also demonstrate that the algorithm behaves as expected, enumerating important initial, intermediate, and final states for HCR in a small box. No unexpected or unintended macrostates were observed.

5.1.4.1 Centroid versus MFE as representative structure

In the Othmer algorithm, the centroid was used as a representative secondary structure. While this typically works well, it is an inconvenient choice because it may not be sampled with high probability. Figure 5.2 shows a pathological example that highlights the disadvantage of using the centroid as a representative structure. For a long enough trajectory, the pair probabilities will converge to the equilibrium values shown in panel a. The two hairpins shown in panel b constitute most of the probability; the centroid structure, containing no base pairs, is sampled at a probability six orders of magnitude lower than either hairpin. If this were used to recognize entry into the macrostate (as is the case in the previous algorithm), few, if any, transitions to the corresponding macrostate would be sampled. To avoid starting from structures that are unlikely to be sampled in the macrostate, we choose to use the sampled minimum free energy structure as the representative structure for each macrostate, instead of using the centroid. To avoid using a single representative stopping structure, we use the macrostate match criterion, relying on base pair distances between sampled pair probability matrices.

5.1.4.2 Base pair distance properties

To determine if a trajectory is at a local equilibrium, the previous window, $\omega_i^{t-\tau_{\text{relax}}:t}$, is constantly compared to the next window $\omega_i^{t:t+\tau_{\text{relax}}}$. We previously described two distance metrics to compare trajectory windows using the distance in variation and the base pair distance. In previous work, the distance in variation was used to recognize transitions between macrostates, while the pair probability distance was used to cluster macrostates [51]. In panels a and b of Figure 5.3, we show

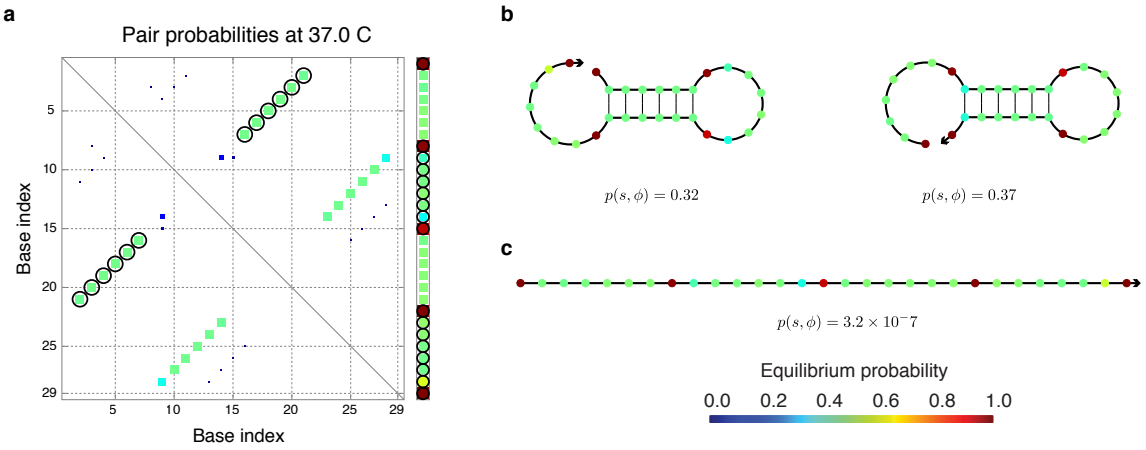


Figure 5.2: Centroid structure versus MFE structure. a) Equilibrium pair probabilities for ϕ = 5′–UCCAGUCGUGGGUGGGACUGGGCACCCAU–3′. Position a, b is shaded with probability $P^{a,b}(\phi)$. b) Two hairpins shown which constitute approximately 70% of the total probability. c) The centroid secondary structure has a probability of being sampled of 3.3×10^{-7}. RNA at 37 °C.

the behavior of each distance metric when adjacent trajectory windows are compared. For these tests, the multistate designer described in Chapter 4 was used to design a sequence to fold into three nearly isoenergetic structures.

For a single trajectory, the distance in variation (panel a) and base pair distance (panel b) were calculated between adjacent time windows between 1 µs and 1000 µs long. Peaks close to unity in the distance in variation are indicative of transitions between macrostates. Peaks show up in the same location in the plot of normalized pair probability, but they do not reach unity. It is clear that the base pair distance converges to zero for shorter windows than the distance in variation, but has lower peaks during transitions. We choose to use the base pair distance for its fast convergence towards zero (which becomes even more apparent for longer sequences) and for its low storage cost.

After an initial macrostate has been discovered, each subsequent trajectory is continuously compared to all known macrostates. These distances are used to determine when a trajectory enters or exits known macrostates. The small box coarse-graining algorithm was used to identify the macrostates for the previously described sequence. The minimum free energy structures for the macrostates are shown in Figure 5.3c. After identifying the macrostates, the windows of the trajectory used in panels a and b were compared to each of the discovered macrostates. The base pair distances to each macrostate over time are shown in Figure 5.3d. The current macrostate and transitions to and from macrostates are clearly visible. The trajectory starts in macrostate A, transitions to B, then back to A, and finally to C. During initial macrostate identification, the trajectory would be stopped as soon as the macrostate match criterion was satisfied, at $t \approx 0$ ms. During macrostate exploration, a trajectory starting from macrostate A, as shown, would stop as soon as it entered another macrostate, e.g., macrostate B at 4.5 ms.

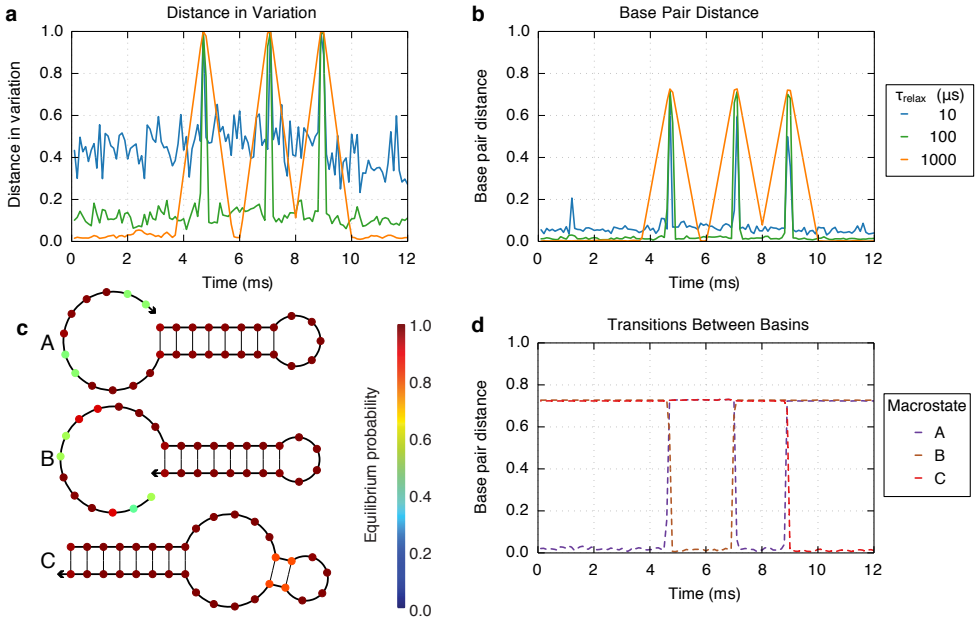


Figure 5.3: Distance metric characterization. a) The distance in variation between the previous and next windows of length τ_{relax} $d^V(\omega_i^{t-\tau_{\text{relax}}:t}, \omega_i^{t:t+\tau_{\text{relax}}})$. b) The base pair distance between the previous and next windows of length τ_{relax}, $d^P(\omega_i^{t-\tau_{\text{relax}}:t}, \omega_i^{t:t+\tau_{\text{relax}}})$, for the same trajectory. c) Representative MFE structures for macrostates identified using small box coarse-graining. Bases are shaded according to their sampled probabilities. d) Base pair distance to each macrostate over time for the same trajectory used in panels a and b. RNA at 25 °C, ϕ = 5′–CCTATAGGAGAAACCTATAGGAAGACCTATAGG–3′, ω^0 = (((((((((.....)))))))))....((....)), $\tau_{\text{relax}} = 100$ µs (panel d), $\tau_{\max} = 10$ s.

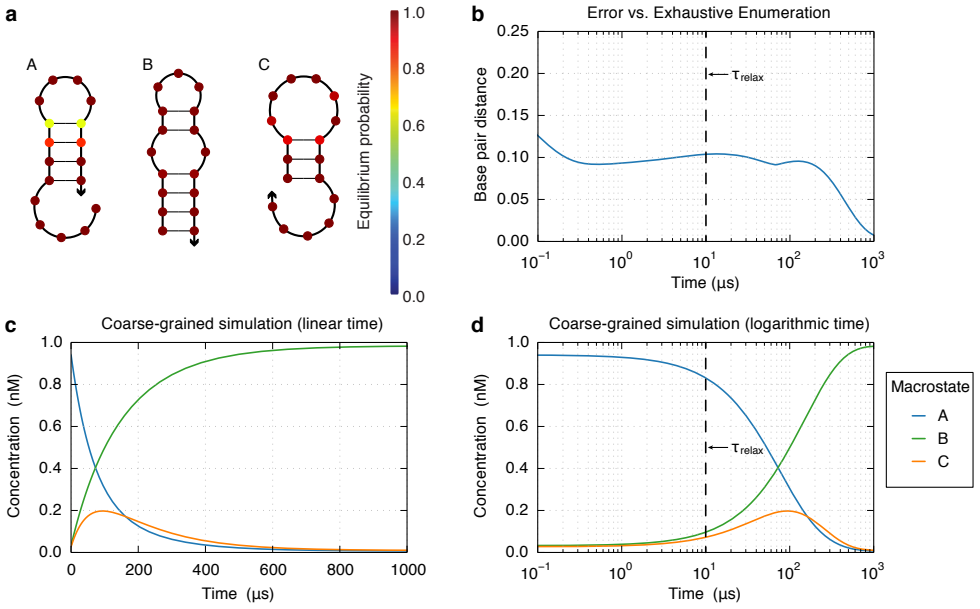


Figure 5.4: Small box coarse-grained results versus exhaustively enumerated results. a) Identified macrostates and transition rates between them. b) Base pair distance between simulations of the full master equation and the coarse grained master equation. c) Master equation simulation, linearly scaled time axis. d) Master equation simulation, logarithmically scaled time axis. RNA at 25 °C, $\phi = 5'$–GCGUCGCGUCGCUAUGC–$3'$, $\omega^0 = \texttt{.....((((....))))}$, $\tau_{\text{relax}} = 10$ μs, $\tau_{\max} = 1$ s.

5.1.4.3 Coarse-grained versus enumerated solution

To check the accuracy of the small box coarse-graining algorithm, we compared a coarse-grained simulation of the base-pair probabilities of a 17-nt sequence to base-pair probabilities calculated using a full simulation of the master equation. The full master equation pair probabilities were calculated using an explicit sum at each timepoint

$$\bar{P}^{a,b}(t) = \sum_{j \in \Omega^*} \bar{p}_j(t) S^{a,b}(s_j). \tag{5.20}$$

The coarse-grained master equation pair probabilities were calculated using a coarse-grained version:

$$\hat{P}^{a,b}(t) = \sum_{h \in \Omega} p_h(t) \tilde{P}_h^{a,b}. \tag{5.21}$$

The base pair distance was calculated at each point in time using

$$\hat{d}^P(\bar{P}^{a,b}(t), \hat{P}^{a,b}(t)) = \sum_{\substack{1 \le a \le |\phi_j| \\ 1 \le b \le |\phi_j|+1}} \left| \bar{P}^{a,b}(t) - \tilde{P}^{a,b}(t) \right|. \tag{5.22}$$

The results are shown in Figure 5.4. Three macrostates were identified (panel a). The base pair distance between the coarse-grained simulation and full simulation quickly drops to 0.1 (10% of

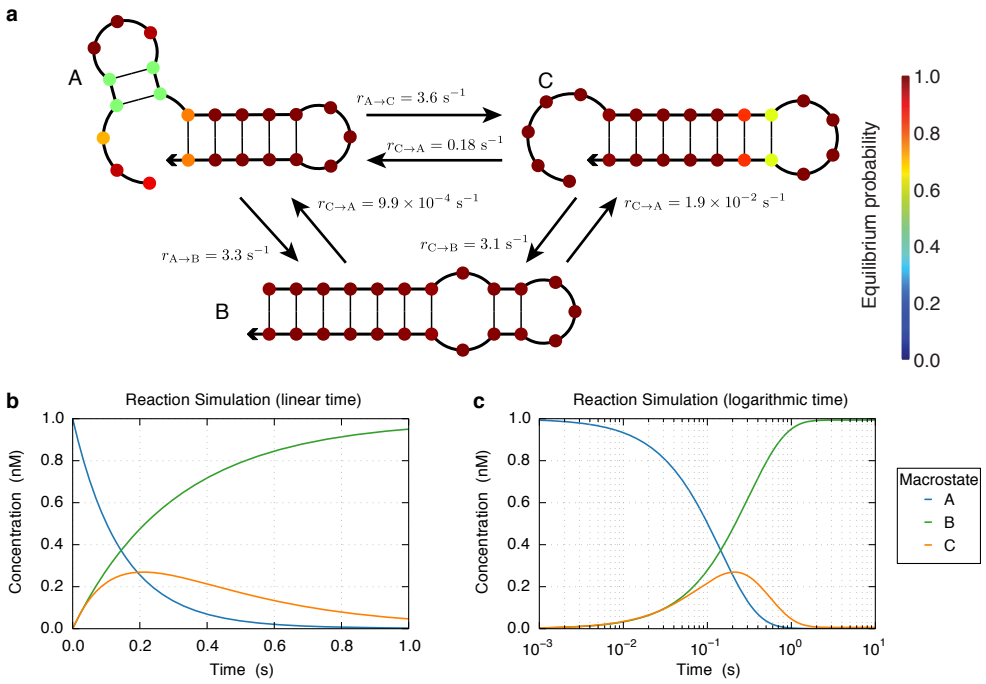


Figure 5.5: Coarse-grained results for 5′–GUCGCGUCGCGUCGCUAUGCGAC–3′. a) Identified macrostates and transition rates between them. b) Master equation simulation, linearly scaled time axis. c) Master equation simulation, logarithmically scaled time axis. The initial microstate was .....((((((...)))))), similar to the representative structure for macrostate A. RNA at 25 °C, $\tau_{\text{relax}} = 100$ µs, $\tau_{\text{max}} = 10$ s.

nucleotides paired differently on average), and drops near the end of the trajectory at 1 ms (panel b). The coarse-grained simulations (panels c and d) show that trajectories typically start in macrostate A and transition to macrostates B and C. By 1 ms, macrostate B dominates most trajectories.

5.1.4.4 Coarse-grained dynamics of a single strand

Figure 5.5 shows the macrostates (panel a) and master equation solution (panels b and c) for another single strand. Starting from a structure with a short hairpin (macrostate A), the system transitions to either a slightly longer hairpin (macrostate C) or a hairpin that spans nearly the whole strand (macrostate B). Eventually, the system transitions to the long hairpin and remains there.

5.1.4.5 Simulation results for HCR

The coarse-graining algorithm was tested on HCR sequences designed using the multistate sequence design algorithm presented in Chapter 4. The macrostates, rates, and master equation simulations for the HCR system are shown in Figure 5.6. As expected, the hairpins add sequentially to the initiator, transitioning from macrostate A to B to C. The addition rates are within an order of magnitude of each other, and the reverse rates differ by approximately an order of magnitude.

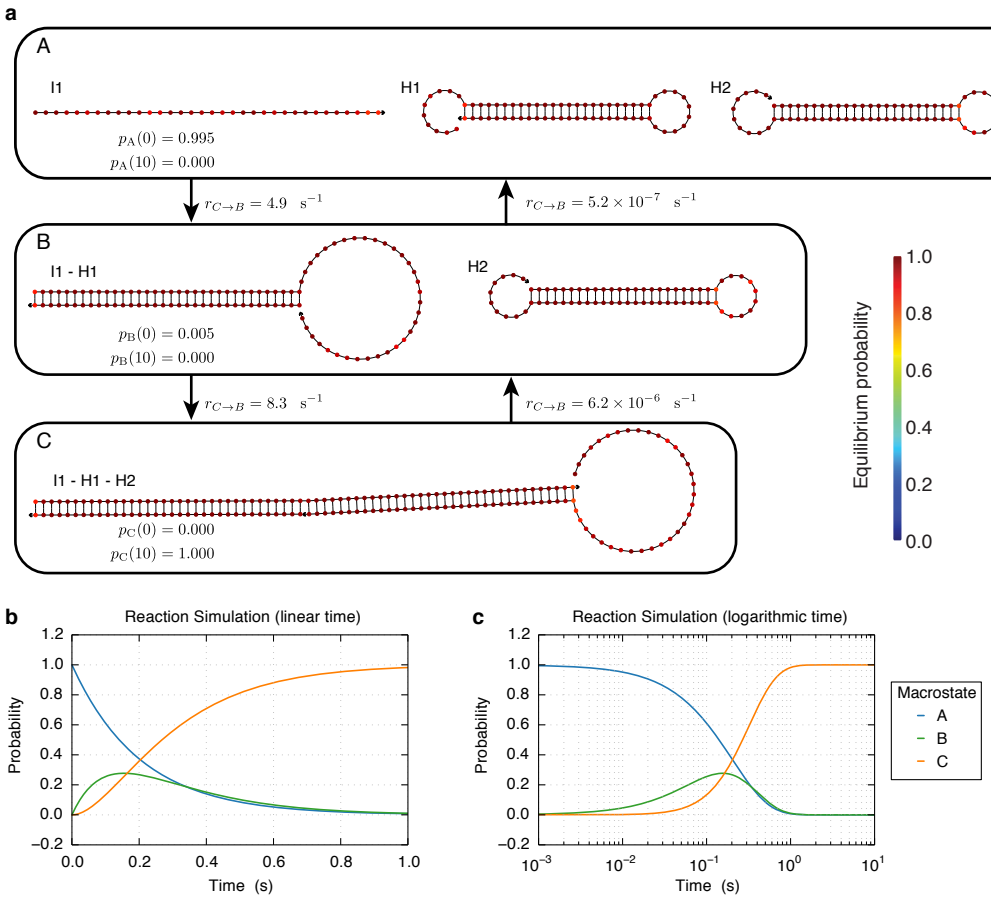


Figure 5.6: HCR small box coarse-graining results. a) Identified macrostates. b) Master equation simulation, linearly scaled time axis. c) Master equation simulation, logarithmically scaled time axis. The initial microstate contained the three strands, I1, H1, and H2, folded into the same structures as the MFE of macrostate A. DNA at 25 °C, 10 µM, $\tau_{\text{relax}} = 2$ ms, $\tau_{\text{max}} = 20$ s.

5.2 Large box coarse-graining

Small box coarse-graining allows coarse-graining of a small number of strands, but it does not address the experimentally relevant problem of determining mass-action kinetics in a test tube. Large box coarse-graining builds on small box coarse-graining to estimate the mass-action kinetics of a test tube of interacting nucleic acid strands. Unimolecular reactions are explored in a process similar to macrostate exploration in the small box. Bimolecular reactions are explored by sampling initial contact states between pairs of reacting species and simulating them until they relax into local equilibria in a process similar to the initial macrostate discovery in small box coarse-graining.

In Section 5.2.1, we provide an overview of the algorithm and introduce key concepts. In Section 5.2.2, we describe the algorithm in detail, following the structure of the small box coarse-graining section. The methods and results sections demonstrate agreement between the small and large box coarse-graining formulations and demonstrate the utility of the algorithm in capturing the kinetics of engineered reaction pathways.

5.2.1 Overview

From a user-provided set of initial complexes, J^0, corresponding initial secondary structures, $s^0_{J^0}$, and corresponding initial concentrations, $x^0_{J^0}$, large box coarse-graining discovers a set of *reacting species*, J, and bimolecular and unimolecular reaction rates between them, k. Reacting species in the large box are analogous to the macrostates of the small box. Each reacting species is indexed by a unique integer h. In contrast to macrostates, however, each reacting species, J_h, consists of a set of secondary structures from the ensemble of a single complex, as defined in Section 2.3. As in the small box, this set of secondary structures is chosen to reach local equilibrium over a relaxation timescale, τ_{relax}. The reaction rates, k, capture dynamics that are important on timescales longer than τ_{relax} but shorter than τ_{max}.

The first part of the algorithm identifies initial reacting species and their initial concentrations. This is performed by simulating trajectories for each initial complex. Each trajectory starts from the specified initial secondary structure, and continues until it reaches a local equilibrium. This local equilibrium is a reacting species, J_h, and it is added to the set of reacting species, J, and a new trajectory is started. This is repeated many times for each initial complex to discover the possible starting reacting species. The fraction of trajectories that end in each reacting species and the initial concentration for each complex are used to estimate an initial concentration for each reacting species for the large box coarse-grained simulation.

For the rest of the algorithm, the set of reacting species, J, is partitioned into an unexplored subset, J^{u}, and an explored subset, J^{x}. Immediately after discovery, reacting species are unexplored. Unexplored reacting species can only be products of reactions. Explored reacting species can be

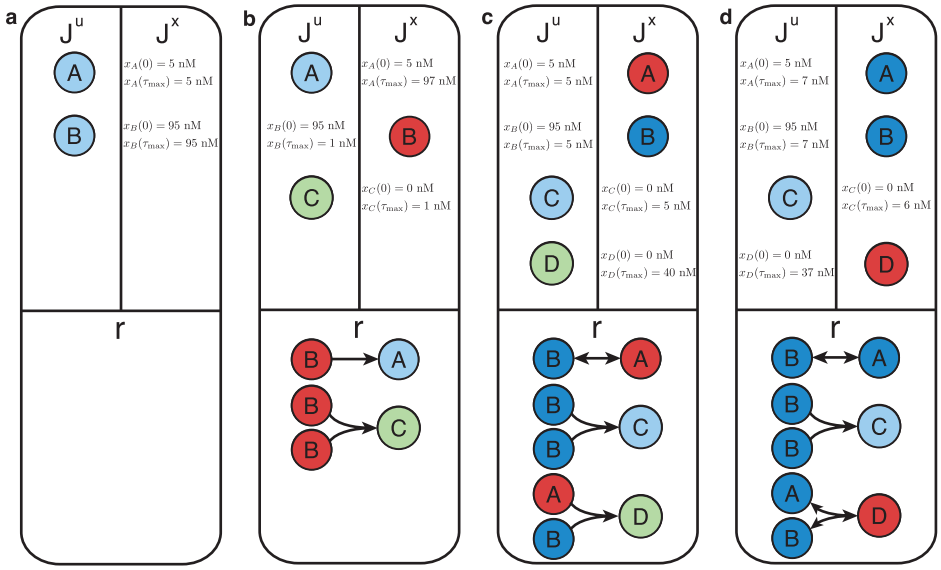


Figure 5.7: Diagram of large box reacting species exploration. a) The algorithm is initialized with the set of complexes, each with an initial structure. Initial reacting species A and B are discovered from these. b) In the first round of reacting species exploration, species B is selected. A unimolecular transition $B \rightarrow A$ and a bimolecular self-reaction is found $B + B \rightarrow C$. c) In the second round, reacting species A is selected, and the reaction $A \rightarrow B$ is discovered via unimolecular rate determination or via detailed balance. A bimolecular reaction to a new species is discovered: $A + B \rightarrow D$. d) In the third iteration, reacting species D is chosen, and the reaction rate for $D \rightarrow A + B$ is estimated either via unimolecular rate exploration or via detailed balance.

either reactants or products and the partition function for each reacting species is estimated. Using this partition function information, detailed balance is enforced for each reaction that acts only on explored reacting species. After discovering initial reacting species (which are added to J^{u}) and estimating their initial concentrations, the rest of the algorithm iteratively selects a reacting species from J^{u}, transfers it to J^{x}, and estimates its partition function, outgoing unimolecular reaction rates, and outgoing bimolecular reaction rates. This iterative exploration is depicted in Figure 5.7. Starting at panel a, reacting species A and B were discovered based on the initial set of complexes and their initial secondary structures. The remaining panels are referenced below.

After estimating the initial concentrations, *reacting species exploration* commences, the reacting species, $J_{\hat{h}}$, containing the highest concentration of nucleotides is transferred to J^{x}, and unimolecular and bimolecular reaction rates are estimated. Unimolecular rates are initially estimated by simulating long trajectories that start in the selected reacting species until the trajectories enter a different reacting species or until the complex dissociates. If the complex dissociates, the two resulting complexes are simulated in isolation until they enter a reacting species. Each resulting product reacting species can be a newly discovered reacting species (discovered when the trajectory reaches local equilibrium outside of its starting reacting species), or it can be a previously discovered reacting species. After simulating $\tau_{\max}$ cumulative time within the selected reacting species,

outgoing unimolecular rates are estimated based on the number of observed transitions from the selected reacting species to each list of product reacting species. This process is very similar to the macrostate exploration process in the small box, but this algorithm isolates complexes upon dissociation.

After estimating unimolecular reactions rates, bimolecular rates are estimated by sampling initial contact states between the selected reacting species and each reacting species in J^{x}, including initial contact states of the selected reacting species with itself. Simulations are started from the sampled initial contact states until the simulation reaches a reacting species (either known or newly discovered) or the complex dissociates (in which case the resulting structures are simulated independently until they individually reach reacting species). Each resulting bimolecular reaction rate, $k_{\{h\}\to\{j\}}$, is estimated based on its estimated collision rate and the fraction of collisions that create its product reacting species $\{j\}$.

At this point, a coarse-grained mass-action simulation can be performed using the current set of reacting species and reaction rates. Using this, the concentrations at the maximum time, $x_h(\tau_{\max})$, for each reacting species, J_h, are estimated. Figure 5.7b shows this stage of the algorithm. Reacting species B was explored since it had the highest initial concentration. A unimolecular reaction was discovered to reacting species A (a previously discovered reacting species) and a bimolecular self-reaction was discovered that resulted in new reacting species C (discovered using the local equilibration criterion).

After the coarse-grained simulation, the unexplored reacting species, $J_{\hat{h}} \in J^{\mathrm{u}}$, with the highest concentration at the end of the simulation is transferred to J^{x}, and its outgoing rates are estimated in a subsequent round of reacting species exploration. In addition to the rate estimates described above, partition function estimates are used to enforce detailed balance for all reactions where the reactants and products are all in J^{x}. Figure 5.7c illustrates the state after the next round of reacting species exploration; bidirectional arrows indicate reactions between members of J^{x}; these rates are calculated to satisfy detailed balance.

The total concentration in unexplored species at time $\tau_{\max}$ is a quantification of the error in the coarse-graining that can be eliminated by further rounds of exploration. Reacting species exploration is repeated, selecting the reacting species containing the highest mass after $\tau_{\max}$ time each iteration, until the fraction of mass in J^{u} is less than a user specified stop fraction f_{stop} of the total mass in the tube. Figure 5.7d illustrates the state after a third round of reacting species exploration. After this round, a single reacting species, C, remains unexplored.

The algorithm is described in the next section and detailed in Algorithm 5.2.

5.2.2 Algorithm

We use adaptations of each ingredient used in small box coarse-graining in the large box algorithm. Reacting species are characterized using similar trajectory criteria, and initial state discovery is again based on short trajectory simulations from user-specified initial conditions, and unimolecular reaction rates are estimated by simulating long trajectories, starting at the representative structure for a reacting species. In addition to these ingredients, we estimate bimolecular reaction rates between reacting species by sampling initial contact secondary structures and simulating short trajectories to find the resulting reacting species.

This section describes each of these ingredients: Section 5.2.2.1 defines the trajectory match criteria for the large box, Section 5.2.2.2 describes how the initial reacting species are discovered and how their initial concentrations are estimated, and Section 5.2.2.3 describes how reacting species are explored, including how unimolecular and bimolecular rates are estimated, how detailed balance is enforced, and how reacting species are chosen for exploration.

At any point during large-box coarse-graining, if the strands can be divided into two complexes that are not connected by any base pairs, they are divided and simulated independently. This ensures that every trajectory in large box coarse-graining always contains a single complex. We continue to write trajectories using ω_i^t notation introduced for the small box for parallelism, but each box consists of a single complex.

5.2.2.1 Characterizing reacting species

Each reacting species, J_h, is characterized by a representative trajectory, $\omega_{J_h}^{0:\tau_{\text{relax}}}$, analogous to a macrostate representative trajectory. Additionally, each reacting species has a representative secondary structure, $\omega_{J_h}^{\text{MFE}}$. Simulations from the reacting species typically start from this secondary structure.

Large box coarse-graining uses similar distance metrics as small box coarse-graining to recognize local equilibria. We define the base pair distance only between trajectories consisting of the same complex, i.e., trajectories with the same strand ordering. Each reacting species consists of secondary structures drawn from a complex, so no comparisons between trajectories containing other complexes are necessary.

At any point in the algorithm, we determine that a reacting species has locally equilibrated when the previous and next windows of length τ_{relax} are approximately the same, indicating the trajectory is approaching ergodicity on this timescale. We characterize this by using (5.8), as before.

In addition to the local equilibration criterion, we wish to be able to determine if a trajectory is in a known reacting species. To do this, we measure the difference between the representative trajectory, $\omega_{J_h}^{0:\tau_{\text{relax}}}$, for each reacting species, J_h, consistent with strand ordering of the current trajectory and

the next window of length τ_{relax} in the current trajectory. Similar to (5.9), a trajectory window, $\omega_i^{t:t+\tau_{\text{relax}}}$, *matches* reacting species Ω_h if they have the same strand ordering and the distance satisfies the *reacting species match criterion*:

$$d^P(\omega_i^{t:t+\tau_{\text{relax}}}, \omega_{\Omega_h}^{0:\tau_{\text{relax}}}) < f_{\text{match}}. \tag{5.23}$$

The reacting species match criterion is used to determine entries to and exits from known reacting species.

5.2.2.2 Identifying initial reacting species

Given an initial set of complexes, J^0, their sequences, ϕ_{J^0}, their initial concentrations, $x_{J^0}^0$, and their initial secondary structures, s_{J^0}, the first step of the algorithm identifies the reacting species each complex relaxes into. For each complex, this is done by simulating M_{init} short trajectories that start at its specified initial secondary structure. The first trajectory is run until the trajectory stabilization criterion (5.8) is met. When this occurs, the window of length τ_{relax} centered on the equilibration time, t, is chosen as a representative trajectory for the new reacting species, J_h:

$$\omega_{J_h}^{0:\tau_{\text{relax}}} \equiv \omega_i^{t-\tau_{\text{relax}}/2:t+\tau_{\text{relax}}/2}. \tag{5.24}$$

Subsequent trajectories halt when they satisfy either the local equilibration criterion or the reacting species match criterion (5.23) with any known reacting species. If the initial complex dissociates before either criterion is met, the dissociated complexes are separated and simulated independently until they individually reach reacting species. This can recur, with the resulting complexes dissociating repeatedly until each complex consists of a single strand. Thus, for each trajectory, the algorithm finds a list of initial reacting species $\{J_h\}$. Each newly discovered reacting species is added to the set of unexplored reacting species, J^{u}.

An initialization counter, $m_{\text{init}}^{j\to\{h\}}$, keeps track of the number of trajectories that start in initial complex J_j^0 that end in the list of reacting species, $\{J_h : h \in \{h\}\}$. After running M_{init} trajectories for each reacting species, the initial concentration of each reacting species, $J_{\hat{h}}$, is estimated as

$$x_{\hat{h}}(0) = \sum_{m_{\text{init}}^{j\to\{h\}}} \sum_{h\in\{h\}} \frac{x_j^0 m_{\text{init}}^{j\to\{h\}} \delta(\hat{h}, h)}{M_{\text{init}}}, \tag{5.25}$$

where the first sum is over all non-zero initialization counters and the second sum is over the list of resulting reacting species corresponding to the current counter, and $\delta(\hat{h}, h)$ is unity if $\hat{h} = h$, and is zero otherwise. Note that the same reacting species can appear multiple times in a list.

5.2.2.3 Reacting species exploration

After identifying initial reacting species and their initial concentrations, the algorithm iteratively explores reacting species, transferring them from J^{u} to J^{x}, and exploring their outgoing reaction rates. Outgoing transition rates are estimated by simulating long Monte Carlo trajectories to find unimolecular reactions, by simulating initial contact states to find bimolecular reactions, and by enforcing detailed balance for reactions where reactants and products have all been explored. Reacting species are selected for exploration based on their concentration during large box coarse-grained simulations. For the first iteration of reacting species exploration, the reacting species, $J_{\hat{h}}$, with the highest initial concentration, $x_{\hat{h}}(0)$, is selected.

Simulating unimolecular reactions. First, unimolecular reactions from the selected reacting species are explored by simulating long trajectories that start at the representative structure for the reacting species. Each trajectory can either a) leave the initial reacting species and enter reacting species in the same complex, b) dissociate into two separate complexes, or c) remain in the initial reacting species until $\tau_{\max}$ cumulative time has been simulated in that reacting species.

The conditions for determining if a trajectory has entered a different reacting species are the same as in small box coarse-graining. A previously discovered reacting species is entered if the reacting species match condition (5.23) is satisfied for any reacting species except the selected reacting species. A new reacting species is discovered if the local equilibration criterion (5.8) is satisfied, and the reacting species match condition is unsatisfied for every known reacting species with the same strand ordering.

If the complex dissociates before either of these conditions are satisfied, the resulting species are simulated in isolation until they reach reacting species (again identified using the reacting species match criterion or the local equilibration criterion). Thus, each observed unimolecular reaction is of the form $J_{\hat{h}} \rightarrow \{J_j : j \in \{j\}\}$, where $\{j\}$ is a list of indices of product reacting species. Counters $m_{\hat{h}\rightarrow\{j\}}$ count the number of times each reaction type is observed.

After $\tau_{\max}$ total time has been simulated, each unimolecular reaction rate $k_{\hat{h},\{j\}}$ is estimated:

$$k_{\hat{h}\rightarrow\{j\}} \equiv \frac{m_{\hat{h}\rightarrow\{j\}}}{\tau_{\max}}. \tag{5.26}$$

Simulating bimolecular reactions. Following estimation of unimolecular reactions, bimolecular reaction rates are estimated. For these, total reaction rates depend on the rate of collision between two molecules and on the fraction of these collisions that result in each list of product species. The rate of collision in the `multistrand` model scales linearly with the number of possible initial contact secondary structures between two molecules. We estimate this using the empirical collision rate k_{bimol} given in Schaeffer's thesis [64] and the average number of possible initial contact states,

based on sampled secondary structures from each reacting species as described below. We estimate the fraction of productive collisions by sampling initial contact secondary structures and observing reacting species they relax to.

For each iteration of reacting species exploration, M_{bimol}, attempts are made to react the chosen reacting species $J_{\hat{h}}$ with each explored reacting species, $J_h \in J^{\text{x}}$, including itself. For each attempt, a trajectory is simulated for τ_{relax} time from the MFE structure of each reacting species, $\omega^{\text{MFE}}_{J_{\hat{h}}}$ and $\omega^{\text{MFE}}_{J_h}$, to generate sampled structures, $\omega^{\text{sample}}_{J_{\hat{h}}}$ and $\omega^{\text{sample}}_{J_h}$, for the trial. For trial i, the number of possible base pairs joining the two structures between exterior loops, $m^{\text{contact}}_{\hat{h}+h,i}$, is recorded and one of these base pairs is sampled to join the complexes, forming secondary structure ω^{i}_{join} with sequence ϕ^{i}_{join}. A trajectory is then simulated, starting from ω^{i}_{join} until it reaches product reacting species $\{j\}$. As in previous cases, this can be either a single product reacting species, or a list of product reacting species (if the initial structure dissociates). Each reacting species may be newly discovered using the local equilibration criterion (5.8) or may have been previously discovered and be recognized via the reacting species match criterion (5.23). The observed reaction is thus of the form $J_{\hat{h}} + J_h \rightarrow \{J_j : j \in \{j\}\}$, where each bimolecular reaction can produce a list of product reacting species. A counter, $m_{\hat{h}+h\rightarrow\{j\}}$, keeps track of the number of times each reaction is observed. Unproductive reactions (reactions that end in the starting pair of reacting species) are ignored.

After simulating M_{bimol} collisions for a pair of reactants, the corresponding reaction rates are estimated. First, the average number of initial contact states,

$$\bar{m}^{\text{contact}}_{\hat{h}+h} \equiv \frac{\sum_{i\in 1,\ldots,M_{\text{bimol}}} m^{\text{contact}}_{\hat{h}+h,i}}{M_{\text{bimol}}} \tag{5.27}$$

is calculated. The rate of collision is then determined using the empirical bimolecular collision rate, k_{bimol}, given in Schaeffer's thesis [64]:

$$k^{\text{contact}}_{\hat{h}+h} \equiv k_{\text{bimol}} \bar{m}^{\text{contact}}_{\hat{h}+h}. \tag{5.28}$$

Using this and the fraction of trials that end in each list of products, we estimate the total bimolecular rate:

$$k_{\hat{h}+h\rightarrow\{j\}} \equiv k^{\text{contact}}_{\hat{h}+h} \frac{\bar{m}_{\hat{h}+h\rightarrow\{j\}}}{M_{\text{bimol}}}. \tag{5.29}$$

Partition function estimation. We estimate the partition function for each reacting species, Q_h, as described in the small box coarse-graining section, except that we use complex free energies instead of small box free energies in the Boltzmann summation (5.13). Additionally, each reacting species has the potential to be symmetric. We account for this by treating all strands as distinguishable at the secondary structure level and then dividing the partition function by the maximum symmetry,

simultaneously correcting for over-counting and rotational symmetry penalties [14].

Enforcing detailed balance. When the reactants and products for a particular reaction have been explored (and thus have estimates of their partition functions), the algorithm enforces detailed balance for the reaction. As in the small box case, the reaction direction that is sampled more often is assumed to be correct and the reverse rate is calculated using detailed balance. In other words, if $m_{\{h\}\to\{j\}} > m_{\{j\}\to\{h\}}$, then the rate calculated using (5.26) or (5.29) is overridden with a rate calculated using detailed balance:

$$k_{\{j\}\to\{h\}} = k_{\{h\}\to\{j\}} \frac{\prod_{h\in\{h\}} Q_h}{\prod_{j\in\{j\}} Q_j}. \tag{5.30}$$

Stop condition After estimating unimolecular and bimolecular rates and satisfying detailed balance within J^{x}, the coarse-grained mass action kinetics are simulated until $\tau_{\max}$. For each species, J_h, this produces an estimate of its concentration at the final time, $x_h(\tau_{\max})$. The final concentration of nucleotides in species in J^{u} at time $\tau_{\max}$ is

$$y_{J^{\mathrm{u}}} \equiv \sum_{h\in J^{\mathrm{u}}} |\phi_h| x_h(\tau_{\max}), \tag{5.31}$$

where $|\phi_h|$ is the number of nucleotides in reacting species J_h. This concentration of nucleotides in unexplored states is a quantification of the deficiency of the current coarse-graining. We choose to use this deficiency to define the stop condition for the coarse-graining procedure:

$$\frac{y_{J^{\mathrm{u}}}}{y_J} \leq f_{\mathrm{stop}}, \tag{5.32}$$

where y_J is the total concentration of nucleotides in the test tube,

$$y_J \equiv \sum_{h\in J} |\phi_h| x_h(0), \tag{5.33}$$

and $f_{\mathrm{stop}} \in [0, 1)$ is a user specified stop fraction. If (5.32) is satisfied after estimating final reacting species concentrations, exploration terminates and the current set of reacting species, their representative trajectories, partition functions, and reaction rates are returned. Otherwise, at least one unexplored reacting species forms at non-zero concentration at time $\tau_{\max}$. The unexplored reacting species with the most mass in the tube at $\tau_{\max}$,

$$J_{\hat{h}} \equiv \underset{J_h\in J^{\mathrm{u}}}{\operatorname{argmax}} \left[x_h(\tau_{\max}) |\phi_h| \right], \tag{5.34}$$

LargeboxCG($J^0, \omega^0_{J^0}, \phi_{J^0}, x^0_{J^0}$)
  $J^{\mathrm{u}}, \omega_J, x_J(0) \leftarrow$ FindInitialStates($\omega(0), \phi, x^0$)
  $x_J(\tau_{\max}) \leftarrow x_J(0)$
  while $\sum_{j \in J^{\mathrm{u}}} x_j(\tau_{\max})|\phi_j| > f_{\mathrm{stop}} \sum_{j \in J^0} x^0_j |\phi_j|$
    $J_{\hat{h}} \leftarrow \mathrm{argmax}_{J_h \in J^{\mathrm{u}}}(x_h(\tau_{\max})|\phi_j|)$
    $J^{\mathrm{x}} \leftarrow J^{\mathrm{x}} \cup \{J_{\hat{h}}\}$
    $J^{\mathrm{u}} \leftarrow J^{\mathrm{u}} \setminus \{J_{\hat{h}}\}$
    $J, \omega_J, m_{\hat{h} \to \{J\}} \leftarrow$
      ExploreUnimolecular($J_{\hat{h}}, \omega_J, J$)
    $J, \omega_J, m_{\hat{h}+J \to \{J\}}, m^{\mathrm{contact}}_{\{J\}} \leftarrow$
      ExploreBimolecular($\hat{h}, \omega_J, J^{\mathrm{u}}, J^{\mathrm{x}}$)
    $Q_{\hat{h}} \leftarrow$ EstimatePfunc($\omega_{J_{\hat{h}}}$)
    $k_{\{J\} \to \{J\}} \leftarrow$
      EstimateRates($m_{\{J\} \to \{J\}}, \bar{m}^{\mathrm{contact}}_{\{J\}}, Q_J$)
    $x_J(\tau_{\max}) \leftarrow$ SimulateCG($x_J, x_J(0), \tau_{\max}$)
  return $J, Q_J, k_{\{J\} \to \{J\}}$

ExploreUnimolecular($J_{\hat{h}}, \omega_J, J$)
  $t \leftarrow 0$
  $\gamma_{\mathrm{product}} = \{\}$
  while $t < \tau_{\max}$
    $\{j\}, t', J, \omega_J \leftarrow$ LongTrajectory($\omega^{\mathrm{MFE}}_{J_{\hat{h}}}, \phi,$
      $J, \omega_J, t - \tau_{\max}$)
    $t \leftarrow t + t'$
    if $\{j\} \notin \gamma_{\mathrm{product}}$
      $\gamma_{\mathrm{product}} \leftarrow \gamma_{\mathrm{product}} \cup \{j\}$
      $m_{\hat{h} \to \{j\}} \leftarrow 1$
    else
      $m_{\hat{h} \to \{j\}} \leftarrow m_{\hat{h} \to \{j\}} + 1$
  return $J^{\mathrm{u}}, \omega_J, m_{\hat{h} \to \{J\}}$

ExploreBimolecular($\hat{h}, \omega_J, J^{\mathrm{x}}, J^{\mathrm{u}}$)
  for $h \in J^{\mathrm{x}}$
    $\gamma_{\mathrm{product}} = \{\}$
    for $i \in \{1, \ldots, M_{\mathrm{bimol}}$
      $\omega^{\mathrm{sample}}_{J_{\hat{h}}} \leftarrow$ ShortTrajectory($\omega^{\mathrm{MFE}}_{J_{\hat{h}}}$)
      $\omega^{\mathrm{sample}}_{J_h} \leftarrow$ ShortTrajectory($\omega^{\mathrm{MFE}}_{J_h}$)
      $m^{\mathrm{contact}}_{\hat{h}+h,i} \leftarrow$ PossiblePairs($\omega^{\mathrm{sample}}_{J_{\hat{h}}}, \omega^{\mathrm{sample}}_{J_h}$)
      $\omega^i_{\mathrm{join}}, \phi^i_{\mathrm{join}} \leftarrow$ RandomJoin($\omega^{\mathrm{sample}}_{J_{\hat{h}}}, \omega^{\mathrm{sample}}_{J_h}$)
      $J, \{j\} \leftarrow$ MatchTrajectory($\omega^i_{\mathrm{join}}, \phi^i_{\mathrm{join}}, \omega_J, J$)
      if $\{j\} \notin \gamma_{\mathrm{product}}$
        $m_{\hat{h}+h \to \{j\}} \leftarrow 1$
      else
        $m_{\hat{h}+h \to \{j\}} \leftarrow m_{\hat{h}+h \to \{j\}} + 1$
    $\bar{m}^{\mathrm{contact}}_{\hat{h},h} \leftarrow \sum_{i \in 1,\ldots,M_{\mathrm{bimol}}} m^{\mathrm{contact}}_{\hat{h},h,i} / M_{\mathrm{bimol}}$
  return $m_{\hat{h}+J \to \{J\}}, \bar{m}^{\mathrm{contact}}_{\hat{h}+J}$

EstimateRates($m_{\{J\} \to \{J\}}, \bar{m}^{\mathrm{contact}}_{\{J\}}, Q_J$)
  for $m_{\{h\} \to \{j\}} \in m_{\{J\} \to \{J\}}$
    if $m_{\{h\} \to \{j\}} > m_{\{j\} \to \{\hat{j}\}}$
      if $|\{h\}| = 1$
        $k_{\{h\} \to \{j\}} \leftarrow m_{\{h\} \to \{j\}} / \tau_{\max}$
      else if $|\{h\}| = 2$
        $k_{\{h\} \to \{j\}} \leftarrow m_{\{h\} \to \{j\}} m^{\mathrm{contact}}_{\{j\}} k_{\mathrm{bimol}} / M_{\mathrm{bimol}}$
      if $\{j\} \subset J^{\mathrm{x}}$
      else
        $k_{\{j\} \to \{\hat{j}\}} \leftarrow 0$
  return $k_{\{J\} \to \{J\}}$

Algorithm 5.2: Large box coarse-graining. Given an initial set of reacting species J^0 and initial concentrations x^0, the function LargeboxCG($J^0, \omega^0_{J^0}, \phi_{J^0}, x^0_{J^0}$) returns the reacting species and reaction rates for a test tube of interacting nucleic acid strands.

is chosen for a new round of reacting species exploration. This is repeated until the stop condition (5.32) is satisfied.

5.2.2.4 Small box versus large box

The structure of the small and large box algorithms is very similar. To expose the parallelism, Table 5.1 compares the key notation and terminology in the small and large box coarse-graining algorithms. In the next subsection, we describe how transition rates in the small box can be compared to unimolecular and bimolecular rates in the large box.

Converting small box rates to large box rates. We can do a basic check of the estimated reaction rates for the large box calculations using the transition rates from small box coarse-graining. For this, we note the equivalence of the two sets of rates [29, 30]. For unimolecular reactions, reaction

Description	Small box	Large box
Species name	*macrostate*	*reacting species*
Species symbol	Ω	J
Unexplored symbol	Ω^{u}	J^{u}
Explored symbol	Ω^{x}	J^{x}
Coarse-grained quantities	*probability*	*concentration*
Quantity symbol	$p_h(t)$	$x_h(t)$
Rates	*transition rate*	*reaction rate*
Rate symbol	$r_{h\to j}$	$k_{\{h\}\to\{j\}}$
Stop condition	$p_{\Omega^{\mathrm{u}}} \leq p_{\mathrm{stop}}$	$x_{J^{\mathrm{u}}}/x_J \leq f_{\mathrm{stop}}$

Table 5.1: Comparison of small and large box quantities.

rates of the large box and transition rates of the small box should be equal:

$$r_{i_1\to i_2} \approx k_{h_1\to h_2}. \tag{5.35}$$

For bimolecular reactions, the large box reaction rate needs to be normalized by Avogadro's constant, N_{A}, and the volume of the small box V_{smallbox}; reaction rates are given in molar concentrations or mole fractions per unit time, but small box transition rates are given in events per unit time. For a transition in the small box, $\Omega_{j_1} \to \Omega_{j_2}$, representing a bimolecular reaction in the large box, $J_{h_1} + J_{h_2} \to J_{h_3}$, we find that

$$r_{i_1\to i_2} = \frac{k_{h_1+h_2\to h_3}}{V_{\mathrm{smallbox}} N_{\mathrm{A}}}, \tag{5.36}$$

where V_{smallbox} is the volume of the small box [30]. Following Schaeffer's definitions [64], we have defined the volume of the small box to be the volume in which a single copy of each strand is present for a user-specified concentration x_{smallbox}:

$$V_{\mathrm{smallbox}} \equiv \frac{1}{N_{\mathrm{A}} x_{\mathrm{smallbox}}}. \tag{5.37}$$

Thus, after coarse-graining using both approaches, we expect to be able to relate the transition rate of the small box to the reaction rate of the large box using

$$k_{h_1+h_2\to h_3} \approx \frac{r_{j_1\to j_2}}{x_{\mathrm{smallbox}}}. \tag{5.38}$$

5.2.3 Methods

The large box coarse-graining algorithm was first tested on a dimerization reaction and the results were compared to the equivalent small box coarse-graining results calculated at several concentrations to check the equivalence of the coarse-graining approaches. These examples were run using RNA at 25 °C with no dangles.

x_{smallbox} (M)	$r_{i_1 \to i_2}$ $(1/s)$	$r_{i_1 \to i_2}/x_{\text{smallbox}}$ $(1/M/s)$
1×10^{-3}	4.10×10^{-4}	0.410
1×10^{-4}	3.99×10^{-5}	0.399
1×10^{-5}	3.92×10^{-6}	0.392
1×10^{-6}	3.90×10^{-7}	0.390
large box $k_{\{j_1,j_2\} \to j_3}$:		0.415

Table 5.2: Small and large box dimerization rates. Small box simulations started from a single copy of each strand 5′–GGGACGAGGC–3′ and 5′–GCCUCGUCCC–3′. The large box simulation was started with 1 µM of each strand in solution. Parameters: $\tau_{\text{relax}} = 100$ µs, $\tau_{\text{max}} = 50$ s, $f_{\text{match}} = 0.1$, $f_{\text{equil}} = 0.02$. For the large box, $M_{\text{bimol}} = 2000$.

The algorithm was then tested on a variety of sequences designed using the multi-state sequence design algorithm: a DNA AND gate, a cooperative gate, and an HCR system. Since HCR is intended to form an infinite polymer, a maximum simulation time of $\tau_{\text{max}} = 10$ s was used during enumeration. These tests were run using DNA at 25 °C with no dangles.

5.2.3.1 Implementation

The algorithm was coded in C, calling the `multistrand` C++ library to implement the secondary structure kinetic model. All graphs were generated using `matplotlib` [35].

5.2.4 Results

5.2.4.1 Simulating dimerization

The dimerization of two strands was simulated using both small and large box coarse-graining. The resulting dimerization rate constants are shown in Table 5.2. We would hope that the reaction rates of the small box would converge to the reaction rate derived using the large-box simulation. Discouragingly, while the rates only differ by 5%, the rates for the small box appear to be converging to a slower rate than the rate calculated using large-box coarse-graining.

5.2.4.2 Simulating logic gates

The logic gate designs were simulated, starting with the gate complex and the two input strands. Off-target formation was rare. The resulting reacting species and mass action kinetics are shown in Figure 5.8. The desired mechanism is followed precisely, and each discovered reacting species has the desired base pairing properties.

5.2.4.3 Simulating cooperative gates

The cooperative gate designs were also simulated. In this case, the algorithm needed to capture the intermediate state when either of the input strands bound to the gate. We see that these

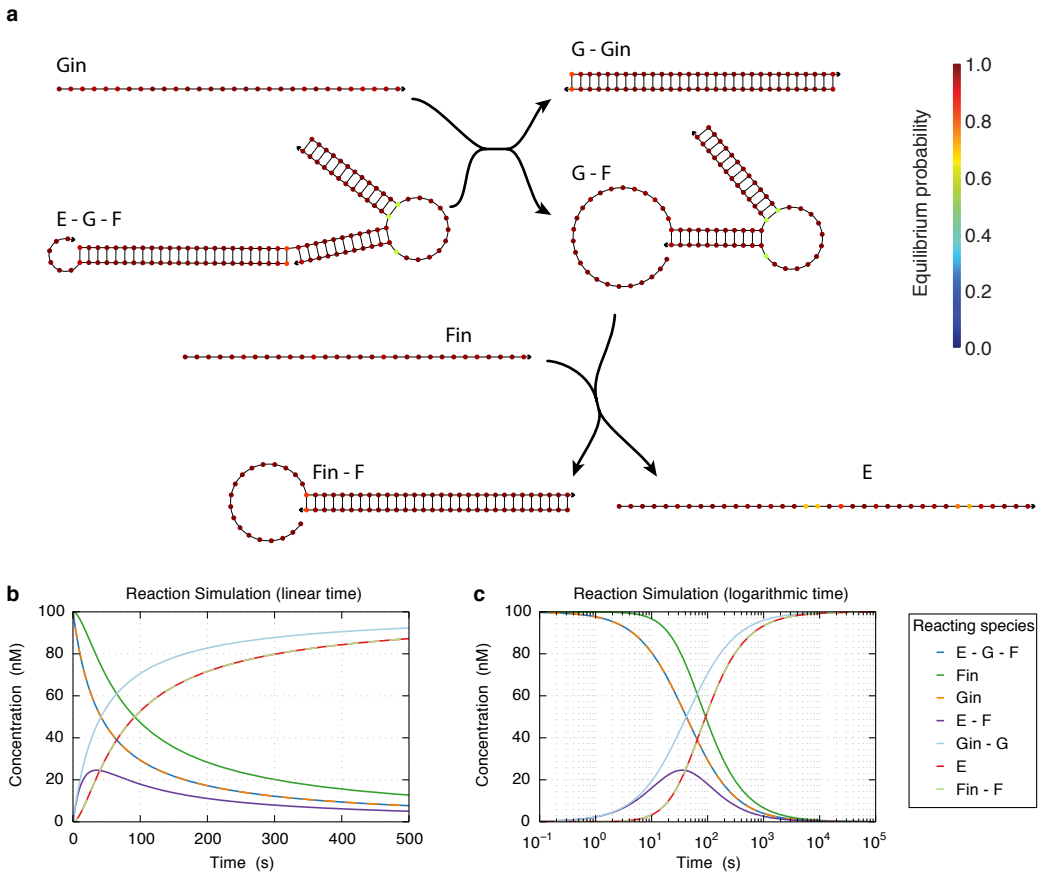


Figure 5.8: Large box coarse-grained results for AND gate [65]. a) Reacting species discovered. b) Reaction simulation with a linear time axis. c) Reaction simulation with a logarithmic time axis. Only enumerated reacting species that form at higher than 1 pM within 10^5 seconds are shown. Parameters: $\tau_{\text{relax}} = 3$ ms, $\tau_{\text{max}} = 100$ s, $f_{\text{match}} = 0.1$, $f_{\text{equil}} = 0.02$, $M_{\text{bimol}} = 2000$.

transients were captured and the intended final product was produced with minimal production of side products. A dimer between the output P and input T1 was predicted to form, but the concentration remained below 1 pM at all times, so it is not shown.

5.2.4.4 Simulating HCR

HCR designs were also simulated using the large box coarse-graining algorithm. Identified reacting species and mass action simulations are shown in Figure 5.10. Unlike small box coarse-graining, the large box simulation could create increasingly long polymers. We ran the mass-action kinetics for only 10 seconds during reacting species exploration to avoid enumerating polymers that were excessively long. Some side products were discovered, but their final concentrations remained below 10 fM for the entire simulation, so they are not shown.

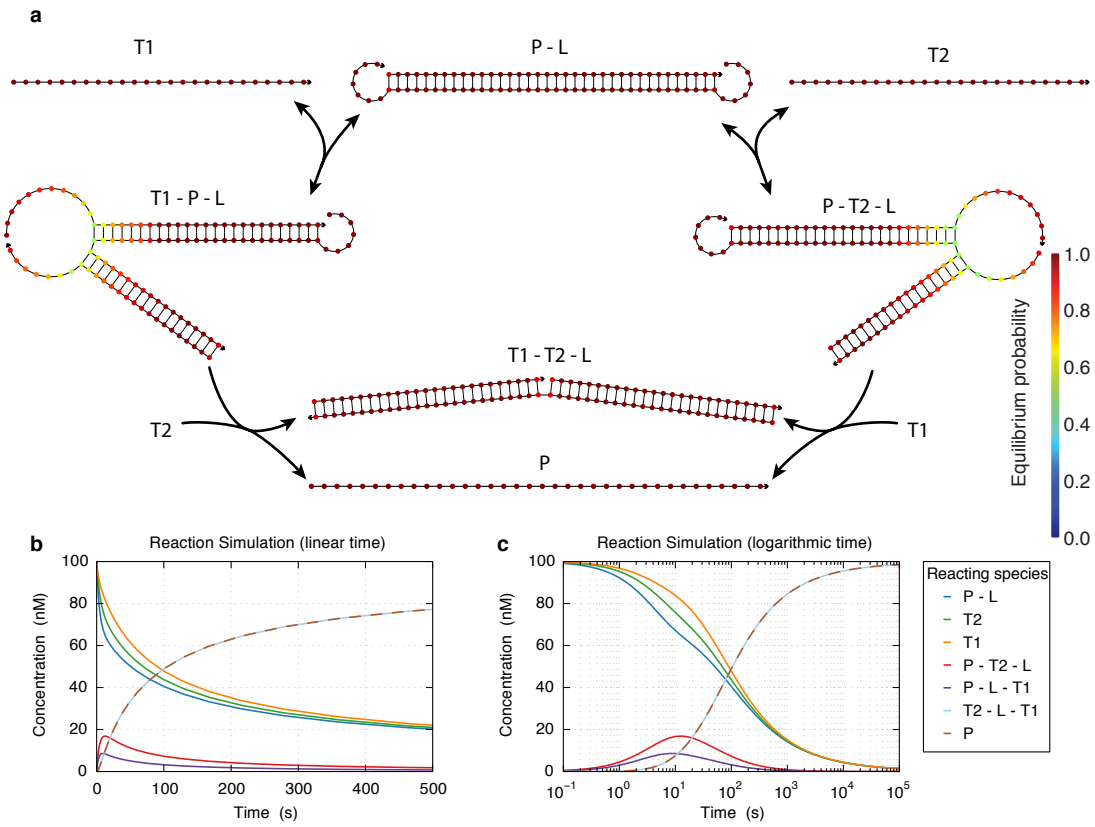


Figure 5.9: Large box coarse-grained results for cooperative gate [84]. a) Reacting species discovered. b) Reaction simulation with a linear time axis. c) Reaction simulation with a logarithmic time axis. Only enumerated reacting species that form at higher than 1 pM within 10^5 seconds are shown. Parameters: $\tau_{\text{relax}} = 3$ ms, $\tau_{\text{max}} = 100$ s, $f_{\text{match}} = 0.1$, $f_{\text{equil}} = 0.02$, $M_{\text{bimol}} = 2000$.

5.3 Limitations

The current approach for coarse-graining has several important limitations that should be addressed in future work. The underlying kinetic model needs to be updated, the relaxation timescale, τ_{relax}, should be automatically chosen or at least validated, and the stopping criteria for both coarse-grainings should be more explicitly characterized, particularly for large box coarse-graining.

The most fundamental of these is the kinetic model. The coarse-grained algorithm cannot be expected to match experimental results if the underlying secondary structure kinetics predict reaction rates that are incorrect by several orders of magnitude. After a model is developed that agrees with experiments, the coarse-graining algorithm can be updated.

The second limitation is the selection of the relaxation timescale, τ_{relax}, by the user. Ideally, the algorithm would discover τ_{relax} based on an initial guess by the user. Currently, if τ_{relax} is chosen such that timescale separation is not achieved, new basins are often spuriously identified. This sometimes results in the enumeration of a large number of nearly-identical macrostates or reacting species. To

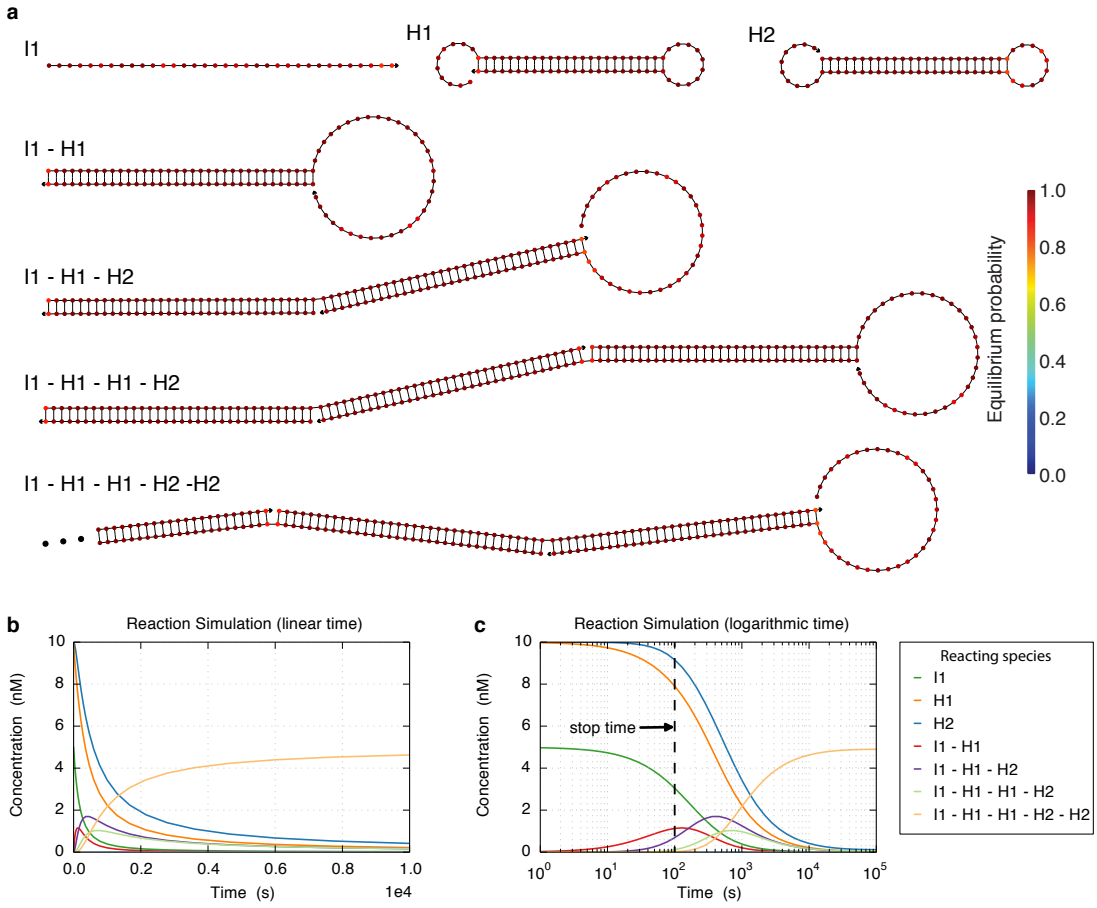


Figure 5.10: Large box results for HCR. a) Reacting species discovered. b) Reaction simulation with a linear time axis. c) Reaction simulation with a logarithmic time axis. Only enumerated reacting species that formed at higher than 1 pM during the full simulation timescale are shown. Mass action results were cut off at 100 s during the coarse-graining algorithm (depicted by the dashed black line in panel c). Parameters: $\tau_{\text{relax}} = 3$ ms, $\tau_{\text{max}} = 10$ s, $f_{\text{match}} = 0.1$, $f_{\text{equil}} = 0.02$, $M_{\text{bimol}} = 2000$.

avoid this situation in practice, it will be necessary to recognize invalid relaxation timescales and either produce a warning or correct the timescale.

In addition to selection of τ_{relax}, the current reliance on a user-specified maximum timescale and stop condition is unsatisfying. It would be beneficial to include known thermodynamic properties as part of the stopping criterion.

Another improvement could take transition times into account. The current bimolecular reaction exploration procedure assumes the time spent relaxing from a bimolecular collision is negligible. Using the approach of section 7.6 of Schaeffer's thesis, these times could be included in rate calculations.

5.4 Conclusions

The coarse-graining approaches discussed in this chapter outline a potential tool for extending secondary-structure level simulations to automatically discover and enumerate macroscopic kinetic properties. For both small and large box coarse-graining, we define an error metric to measure the mass (or probability) lost to unexplored species that have no outgoing reactions (or transitions). Iteratively exploring species, the algorithms systematically discover kinetically important macrostates and transition rates for the small box, and important reacting species and reaction rates for the large box. We compared each algorithm to finer-grained simulations and tested their effectiveness at identifying states in engineered sequences. These algorithms demonstrate viable approaches to exploring the kinetic properties of engineered sequences. With further development, they can become practical tools for nucleic acid engineers to verify kinetic sequence properties prior to strand synthesis.

5.5 Appendix and archive content

Appendix D contains an brief explanation of the deficiency of the large-box stop criterion and a description of the input files contained in the supplementary archive.

Chapter 6

Conclusion

The algorithms developed in this thesis are powerful new tools for designing and analyzing nucleic acid reaction pathways. It is now possible to optimize the thermodynamics of multiple dilute solutions subject to a variety of combinatorial sequence constraints, thus enabling thermodynamic optimization of nucleic acid systems of practical interest.

First, we formulated sequence design of a test tube of interacting nucleic acids as an optimization problem, minimizing the average concentration of nucleotides in the incorrect state at equilibrium over the ensemble of a test tube. Using test tube ensemble focusing and hierarchical ensemble decomposition, we demonstrate efficient design of test tubes containing sixteen strands and thousands of off-targets.

Next, we extended this formulation to the simultaneous design of multiple dilute solutions that share sequence information. This algorithm uses the same ingredients as test tube design to efficiently optimize pathways of interacting nucleic acid strands via the design of initial, intermediate, and final states. To address practical needs of nucleic acid engineers, we extended the sequence initialization and mutation procedures to handle combinatorial sequence constraints during the design process.

Finally, we demonstrated small and large box coarse-graining. These algorithms estimate macroscopic kinetic properties for solutions of interacting nucleic acid strands. They can be used to validate kinetic properties of sequences designed using the thermodynamic design algorithms prior to strand synthesis.

Bibliography

[1] Rosalía Aguirre-Hernández, Holger H. Hoos, and Anne Condon. Computational RNA secondary structure design: empirical complexity and improved methods. *BMC Bioinformatics*, 8:34, January 2007.

[2] Peter B. Allen, Xi Chen, and Andrew D. Ellington. Spatial Control of DNA Reaction Networks by DNA Sequence. *Molecules*, 17(11):13390–402, January 2012.

[3] Mirela Andronescu, Anthony P. Fejes, Frank Hutter, Holger H. Hoos, and Anne Condon. A new algorithm for RNA secondary structure design. *Journal of Molecular Biology*, 336(3):607–24, February 2004.

[4] Mirela Andronescu, Zhi C. Zhang, and Anne Condon. Secondary structure prediction of interacting RNA molecules. *Journal of Molecular Biology*, 345:987–1001, 2005.

[5] Assaf Avihoo, Alexander Churkin, and Danny Barash. RNAexinv: An extended inverse RNA folding from shape and physical attributes to sequences. *BMC Bioinformatics*, 12, 2011.

[6] Jonathan Bath and Andrew J. Turberfield. DNA nanomachines. *Nature nanotechnology*, 2:275–284, 2007.

[7] Sanchita Bhadra and Andrew D. Ellington. Design and application of cotranscriptional non-enzymatic RNA circuits and signal transducers. *Nucleic Acids Research*, pages 1–16, 2014.

[8] Bernd Burghardt and Alexander K. Hartmann. RNA secondary structure design. *Physical Review E*, 75:21920, 2007.

[9] Anke Busch and Rolf Backofen. INFO-RNA–a fast approach to inverse RNA folding. *Bioinformatics*, 22(15):1823–31, August 2006.

[10] Thomas E. Cheatham and Matthew A. Young. Simulation of Nucleic Acids: Successes, Limitations, and Promise. *Biopolymers*, 56(2001):232–256, 2000.

[11] Harry M. T. Choi, Victor A. Beck, and Niles A. Pierce. Next-Generation in Situ Hybridization Chain Reaction: Higher Gain, Lower Cost, Greater Durability. *ACS Nano*, 2014.

[12] Harry M. T. Choi, Joann Y. Chang, Le A. Trinh, Jennifer E. Padilla, Scott E. Fraser, and Niles A. Pierce. Programmable in situ amplification for multiplexed imaging of mRNA expression. *Nature Biotechnology*, 28:1208–1212, 2010.

[13] Rina Dechter. *Constraint Processing*. Morgan Kaufmann, San Francisco, 2003.

[14] Robert M. Dirks, Justin S. Bois, Joseph M. Schaeffer, Erik Winfree, and Niles A. Pierce. Thermodynamic Analysis of Interacting Nucleic Acid Strands. *SIAM Review*, 49(1):65, 2007.

[15] Robert M. Dirks, Milo Lin, Erik Winfree, and Niles A. Pierce. Paradigms for computational nucleic acid design. *Nucleic Acids Research*, 32(4):1392–1403, January 2004.

[16] Robert M. Dirks and Niles A. Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of Computational Chemistry*, 24(13):1664–77, October 2003.

[17] Robert M. Dirks and Niles A. Pierce. Triggered amplification by hybridization chain reaction. *Proceedings of the National Academy of Sciences of the United States of America*, 101(43):15275–78, 2004.

[18] Ivan Dotu, William A. Lorenz, Pascal Van Hentenryck, and Peter Clote. Computing folding pathways between RNA secondary structures. *Nucleic Acids Research*, 38(5):1711–22, March 2010.

[19] Shawn M. Douglas, Hendrik Dietz, Tim Liedl, Björn Högberg, Franziska Graf, and William M. Shih. Self-assembly of DNA into nanoscale three-dimensional shapes. *Nature*, 459(7245):414–8, May 2009.

[20] Constantine G. Evans and Erik Winfree. DNA Sticky End Design and Assignment for Robust Algorithmic Self-assembly. *DNA Computing and Molecular Programming*, 1:61–75, 2013.

[21] Christoph Flamm, Walter Fontana, Ivo L. Hofacker, and Peter Schuster. RNA folding at elementary step resolution. *RNA*, 6:325–338, 2000.

[22] Christoph Flamm, Ivo L. Hofacker, Sebastian Maurer-stroh, Peter F. Stadler, and Martin Zehl. Design of multistable RNA molecules. *RNA*, 7:254–265, 2001.

[23] Christoph Flamm, Ivo L. Hofacker, Peter F. Stadler, and Michael T. Wolfinger. Barrier Trees of Degenerate Landscapes. *Zeitschrift fuer Physikalische Chemie (Munich)*, 216:155–173, 2002.

[24] James Z. M. Gao, Linda Y. M. Li, and Christian M. Reidys. Inverse folding of RNA pseudoknot structures. *Algorithms for Molecular Biology*, 5(27), 2010.

[25] Juan Antonio Garcia-Martin, Peter Clote, and Ivan Dotu. RNAiFOLD: a constraint programming algorithm for RNA inverse folding and molecular design. *Journal of Bioinformatics and Computational Biology*, 11(2):1350001, April 2013.

[26] Michel Gendreau and Jean-Yves Potvin, editors. *Handbook of Metaheuristics*. Springer, New York, 2nd edition, 2010.

[27] Anirban Ghosh and Manju Bansal. A glossary of DNA structures from A to Z. *Acta Crystallographica Section D: Biological Crystallography*, 59(4):620–626, March 2003.

[28] Alison L. Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International Statistical Review*, 70(3):419, December 2002.

[29] Daniel T. Gillespie. General method for numerically simulating stochastic time evolution of coupled chemical-reactions. *J. Comput. Phys.*, 22(4):403–434, 1976.

[30] Daniel T. Gillespie. Exact Stochastic Simulation of Coupled Chemical Reactions. *The Journal of Physical Chemistry*, 81(25):2340–61, 1977.

[31] Russell P. Goodman, Mike Heilemann, Sören Doose, Christoph M. Erben, Achillefs N. Kapanidis, and Andrew J. Turberfield. Reconfigurable, braced, three-dimensional DNA nanostructures. *Nature Nanotechnology*, 3(2):93–6, February 2008.

[32] Alan C. Hindmarsh. ODEPACK, A Systematized Collection of ODE Solvers. *IMACS Transactions on Scientific Computation*, 1:55–64, 1983.

[33] Lisa M. Hochrein, Maayan Schwarzkopf, Mona Shahgholi, Peng Yin, and Niles A. Pierce. Conditional Dicer substrate formation via shape and sequence transduction with small conditional RNAs. *Journal of the American Chemical Society*, 135(46):17322–30, November 2013.

[34] Ivo L. Hofacker, Walter Fontana, Peter F. Stadler, L. Sebastian Bonhoeffer, Manfred Tacker, and Peter Schuster. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie / Chemical Monthly*, 125(2):167–188, 1994.

[35] John D. Hunter. Matplotlib: A 2D graphics environment. *Computer Science & Engineering*, 9(3):90–95, 2007.

[36] Farren J. Isaacs, Daniel J. Dwyer, Chunming Ding, Dmitri D. Pervouchine, Charles R. Cantor, and James J. Collins. Engineered riboregulators enable post-transcriptional control of gene expression. *Nature Biotechnology*, 22(7):841–847, July 2004.

[37] Eric Jones, Travis Oliphant, Pearu Peterson, and Others. SciPy: Open source scientific tools for Python, 2001.

[38] Kyozi Kawasaki. Diffusion constants near the critical point for time-dependent Ising models. I. *Physical Review*, 783:224–230, 1966.

[39] Yonggang Ke, Luvena L. Ong, William M. Shih, and Peng Yin. Three-dimensional structures self-assembled from DNA bricks. *Science*, 338(6111):1177–83, November 2012.

[40] Ryan T. Koehler and Nicolas Peyret. Thermodynamic Properties of DNA Sequences: Characteristic Values for the Human Genome. *Bioinformatics*, 21(16):3333–3339, 2005.

[41] Marcel Kucharík, Ivo L. Hofacker, Peter F. Stadler, and Jing Qin. Basin Hopping Graph: A computational framework to characterize RNA folding landscapes. *Bioinformatics*, pages 1–8, March 2014.

[42] Richard Lavery, Krystyna Zakrzewska, David Beveridge, Thomas C. Bishop, David A. Case, Thomas Cheatham, Surjit Dixit, B. Jayaram, Filip Lankas, Charles Laughton, John H. Maddocks, Alexis Michon, Roman Osman, Modesto Orozco, Alberto Perez, Tanya Singh, Nada Spackova, and Jiri Sponer. A systematic molecular dynamics study of nearest-neighbor effects on base pair and base pair step conformations and fluctuations in B-DNA. *Nucleic Acids Research*, 38(1):299–313, January 2010.

[43] Alex Levin, Mieszko Lis, Yann Ponty, Charles W. O'Donnell, Srinivas Devadas, Bonnie Berger, and Jérôme Waldispühl. A global sampling approach to designing and reengineering RNA secondary structures. *Nucleic Acids Research*, 40(20):10041–52, November 2012.

[44] Ronny Lorenz, Stephan H. Bernhart, Christian Höner Zu Siederdissen, Hakim Tafer, Christoph Flamm, Peter F. Stadler, and Ivo L. Hofacker. ViennaRNA Package 2.0. *Algorithms for Molecular Biology*, 6:26, January 2011.

[45] Kyle Lund, Anthony J. Manzo, Nadine Dabby, Nicole Michelotti, Alexander Johnson-Buck, Jeanette Nangreave, Steven Taylor, Renjun Pei, Milan N. Stojanovic, Nils G. Walter, Erik Winfree, and Hao Yan. Molecular robots guided by prescriptive landscapes. *Nature*, 465(7295):206–10, May 2010.

[46] Rune B. Lyngsø, James W. J. Anderson, Elena Sizikova, Amarendra Badugu, Tomas Hyland, and Jotun Hein. Frnakenstein: multiple target inverse RNA folding. *BMC Bioinformatics*, 13:260, 2012.

[47] David H. Mathews, Matthew D. Disney, Jessica L. Childs, Susan J. Schroeder, Michael Zuker, and Douglas H. Turner. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proceedings of the National Academy of Sciences of the United States of America*, 101(19):7287–92, May 2004.

[48] David H. Mathews, Jeffrey Sabina, Michael Zuker, and Douglas H. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *Journal of Molecular Biology*, 288(5):911–40, May 1999.

[49] Marco C. Matthies, Stefan Bienert, and Andrew E. Torda. Dynamics in Sequence Space for RNA Secondary Structure Design. *Journal of Chemical Theory and Computation*, 8(10):3663–3670, 2012.

[50] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087, 1953.

[51] Jonathan A. Othmer. *Algorithms for Mapping Nucleic Acid Free Energy Landscapes*. PhD thesis, California Institute of Technology, 2009.

[52] Thomas E. Ouldridge, Rollo L. Hoare, Ard A. Louis, Jonathan P. K. Doye, Jonathan Bath, and Andrew J. Turberfield. Optimizing DNA Nanotechnology through Coarse-Grained Modeling: A Two-Footed DNA Walker. *ACS Nano*, 2013.

[53] Thomas E. Ouldridge, Ard A. Louis, and Jonathan P. K. Doye. DNA Nanotweezers Studied with a Coarse-Grained Model of DNA. *Physical Review Letters*, 104(17):178101, April 2010.

[54] Richard Owczarzy, Bernardo G. Moreira, Yong You, Mark A. Behlke, and Joseph A. Walder. Predicting stability of DNA duplexes in solutions containing magnesium and monovalent cations. *Biochemistry*, 47(19):5336–53, May 2008.

[55] Adrien Padirac, Teruo Fujii, André Estévez-Torres, and Yannick Rondelez. Spatial waves in synthetic biochemical networks. *Journal of the American ...*, 135:14586–14592, 2013.

[56] Nicolas Peyret. *Prediction of nucleic acid hybridization: parameters and algorithms*. PhD thesis, Wayne State University, 2000.

[57] Andre V. Pinheiro, Dongran Han, William M. Shih, and Hao Yan. Challenges and opportunities for structural DNA nanotechnology. *Nature Nanotechnology*, 6(12):763–772, 2011.

[58] Effirul I. Ramlan and Klaus-Peter Zauner. Design of interacting multi-stable nucleic acids for molecular information processing. *Biosystems*, 105(1):14–24, 2011.

[59] Paul W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, 2006.

[60] Frank Ruskey, Carla Savage, and Terry M. Y. Wang. Generating necklaces. *Journal of Algorithms*, 13(3):414–430, 1992.

[61] Natalie Saini, Yu Zhang, Karen Usdin, and Kirill S. Lobachev. When secondary comes first–the importance of non-canonical DNA structures. *Biochimie*, 95(2):117–23, February 2013.

[62] John SantaLucia, Jr. A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proceedings of the National Academy of Sciences of the United States of America*, 95(4):1460–5, February 1998.

[63] John SantaLucia, Jr. and Donald Hicks. The thermodynamics of DNA structural motifs. *Annual Review of Biophysics and Biomolecular Structure*, 33:415–40, January 2004.

[64] Joseph M. Schaeffer. *Stochastic Simulation of the Kinetics of Multiple Interacting Nucleic Acid Strands*. PhD thesis, California Institute of Technology, 2013.

[65] Georg Seelig, David Soloveichik, David Yu Zhang, and Erik Winfree. Enzyme-free nucleic acid logic circuits. *Science*, 314(5805):1585–8, December 2006.

[66] Nadrian C. Seeman. Structural DNA nanotechnology: growing along with Nano Letters. *Nano letters*, 10(6):1971–8, June 2010.

[67] Nadrian C. Seeman and Neville R. Kallenbach. Design of Immobile Nucleic Acid Junctions. *Biophysical journal*, 44:201–209, 1983.

[68] Evan Senter, Ivan Dotu, and Peter Clote. RNA folding pathways and kinetics using 2D energy landscapes. *Journal of mathematical biology*, February 2014.

[69] Martin J. Serra and Douglas H. Turner. Predicting thermodynamic properties of RNA. *Methods in Enzymology*, 259(1977):242–261, 1995.

[70] Michael R Shortreed, Seo Bong Chang, DongGee Hong, Maggie Phillips, Bridget Campion, Dan C Tulpan, Mirela Andronescu, Anne Condon, Holger H Hoos, and Lloyd M Smith. A thermodynamic approach to designing structure-free combinatorial DNA word sets. *Nucleic Acids Research*, 33(15):4965–77, January 2005.

[71] Wenjie Shu, Ming Liu, Hebing Chen, Xiaochen Bo, and Shengqi Wang. ARDesigner: A web-based system for allosteric RNA design. *Journal of Biotechnology*, 150(4):466–473, 2010.

[72] Niranjan Srinivas, Thomas E. Ouldridge, Petr Sulc, Joseph M. Schaeffer, Bernard Yurke, Ard A. Louis, Jonathan P. K. Doye, and Erik Winfree. On the biophysics and kinetics of toehold-mediated DNA strand displacement. *Nucleic Acids Research*, 41(22):10641–58, December 2013.

[73] Akito Taneda. MODENA: a multi-objective RNA inverse folding. *Advances and Applications in Bioinformatics and Chemistry*, 4:1–12, January 2011.

[74] Akito Taneda. Multi-objective genetic algorithm for pseudoknotted RNA sequence design. *Frontiers in Genetics*, 3(36), January 2012.

[75] Nicolaas Godfried van Kampen. *Stochastic Processes in Physics and Chemistry*. North-Holland, 1992.

[76] Petr Šulc, Thomas E. Ouldridge, Flavio Romano, Jonathan P. K. Doye, and Ard A. Louis. Simulating a burnt-bridges DNA motor with a coarse-grained DNA model. *Natural Computing*, pages 1–12, August 2013.

[77] Adam J. M. Wollman, Carlos Sanchez-Cano, Helen M. J. Carstairs, Robert A. Cross, and Andrew J. Turberfield. Transport and self-organization across different length scales powered by motor proteins and programmed by DNA. *Nature nanotechnology*, 9(1):44–7, January 2014.

[78] Stefan Wuchty, Walter Fontana, Ivo L. Hofacker, and Peter Schuster. Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers*, 49(2):145–165, 1999.

[79] Peng Yin, Harry M. T. Choi, Colby R. Calvert, and Niles A. Pierce. Programming biomolecular self-assembly pathways. *Nature*, 451(7176):318–22, January 2008.

[80] Bernard Yurke, Andrew J. Turberfield, Allen P. Mills Jr., Friedrich C. Simmel, and Jennifer L. Neumann. A DNA-fuelled molecular machine made of DNA. *Nature*, 406(6796):605–608, 2000.

[81] Joseph N. Zadeh. *Algorithms for Nucleic Acid Sequence Design*. PhD thesis, California Institute of Technology, 2010.

[82] Joseph N. Zadeh, Conrad D. Steenberg, Justin S. Bois, Brian R. Wolfe, Marshall B. Pierce, Asif R. Khan, Robert M. Dirks, and Niles A. Pierce. NUPACK: Analysis and design of nucleic acid systems. *Journal of Computational Chemistry*, 32(1):170–173, January 2011.

[83] Joseph N. Zadeh, Brian R. Wolfe, and Niles A. Pierce. Nucleic Acid Sequence Design via Efficient Ensemble Defect Optimization. *Journal of Computational Chemistry*, 32(3):439–52, February 2011.

[84] David Yu Zhang. Cooperative hybridization of oligonucleotides. *Journal of the American Chemical Society*, 133(4):1077–86, February 2011.

[85] David Yu Zhang. Towards domain-based sequence design for DNA strand displacement reactions. *DNA Computing and Molecular Programming*, pages 162–175, 2011.

[86] David Yu Zhang and Georg Seelig. Dynamic DNA nanotechnology using strand-displacement reactions. *Nature Chemistry*, 3(2):103–13, February 2011.

Appendix A

Useful algorithms

This appendix contains a few algorithms that were either used as subroutines for the main part of the thesis, or were used as inspiration for this work.

A.1 Necklace generation

Generating the set of possible complexes is easy to do in a naive fashion, where each possible permutation of a given length is generated, and rotationally equivalent complexes are eliminated by some post-processing step. Instead, one can use the FKM algorithm [60] to efficiently enumerate this set. The computational cost of either method is low, but this algorithm is systematic and elegant. The FKM algorithm is detailed in Algorithm A.1.

A.2 Engineered test set generation

In Chapters 3 and 4, the engineered test set was generated by creating randomized trees. Starting from an empty exterior loop, in each iteration, a loop was selected and a stem terminating in a hairpin was appended to the loop. The chosen loop was then extended by an unstructured region. This process was repeated until the structure reached the desired size. This structure was then cut by inserting a nick halfway through. At this point, the algorithm checked feasibility of design by ensuring that the two strands were joined by at least M_{cut} base-pairs and all hairpins were at least 3-nt long (hairpins smaller 3-nt allowed during structure generation so that small interior loops and multiloops are allowed). The engineered structure generation procedure is detailed in Algorithm A.2.

A.3 Branch and propagate

We used a branch-and-propagate algorithm to find feasible sequences during multistate sequence design in Chapter 4. At each step in a branch-and-propagate algorithm, an uninstantiated variable

```
MakeNecklaces(n, k)
    π̂^i ← 0 ∀i ∈ 1, ..., n
    π_* ← {}
    h ← 0
    i ← n
    while i ≠ 0
        π̂^i ← π̂^i + 1
        for j ∈ 1, ..., n − i
            π̂^{i+j} ← π̂^j
        if n mod i = 0
            h ← h + 1
            π_h ← π̂
        i ← n
        while i > 0 and π̂^i ≠ k − 1
            i ← i − 1
    return {π_1, ..., π_h}
```

Algorithm A.1: Algorithm to enumerate all necklaces, i.e., all complexes. Given the number of strand types k and a length n, this algorithm returns the set of all sequences of length n, treating all circular permutations as equivalent. By calling this for every length from 1 to $L^{\max}$, it is possible to enumerate all possible complexes in a test tube.

$\hat{v}_i$ was chosen based on a heuristic. One value from its domain $\mathcal{D}_{i,j}$ was chosen, again based on a heuristic. The variable was instantiated to that value and all other values removed from its domain [1]. Constraints were propagated by removing values from the domains of the remaining uninstantiated variables. Values were removed if no consistent partial instantiation could be made that includes both the current instantiation set and instantiating the variable to that value. Full consistency of each value was infeasible to check, so local consistency that checks consistency with respect to a small subset of constraints simultaneously was typically used. The basic branch and propagate algorithm is shown in Algorithm A.3.

Typically, the procedure to perform mutations in multistate design preferentially selects instantiations that have a minimal number of nucleotides changed from the original sequence. One could also use branch-and-bound to find the mutation that changes the fewest nucleotides, but this actually provides no benefit in design quality and takes substantially longer per mutation (data not shown). If the constraint propagation algorithm is replaced this may become feasible.

A.4 Iterated local search

One of the primary advantages of the design formulations that were shown in Chapter 3 and Chapter 4 was use of an explicit defect estimate at each level of the decomposition forest. Throughout

[1] Other algorithms split the domain into several segments. This is valuable for linear constraints or real-valued constraints (e.g. in mixed integer programming). It is less helpful for the pure combinatorial constraints we use in this algorithm.

each design algorithm, we construct and gradually improve the quality of these defect estimates. One potential advantage of this is that the current strategy for leaf optimization can be replaced with any search metaheuristic. We use a form of iterated local search in both algorithms, but this could just as easily be replaced with tabu search, simulated annealing, genetic algorithms, or other metaheuristics [26]. To see the similarity between iterated local search and the OptimizeLeaves procedure, it may be helpful to write down the general form of iterated local search, as shown in Algorithm A.4. Compare this to the OptimizeLeaves procedure in Algorithm 3.1 to see the similarity.

```
EngineeredStructure(N)
    M_cut ← 8
    s ← EngineeredTrial(N)
    while MinCut(s) < M_cut or MinHairpin(s) < 3
        s ← EngineeredTrial(N)
    return s

EngineeredTrial(N)
    H_min ← 4
    H_max ← 16
    U_min ← 0
    U_max ← 8
    l ← []
    l.append(Unpaired(UniformSample(U_min, U_max)))
    ŝ ← [l_0]
    while |ŝ| < N
        l ← SampleLoop(ŝ)
        m ← N − |ŝ|
        if m < 2H_min + U_min
            l ← l | Unpaired(m)
        else
            H ← UniformSample(H_min, min(H_max, m/2))
            U_1 ← UniformSample(U_min, min(U_max, m − 2H))
            U_2 ← UniformSample(U_min, min(U_max, m − 2H − U_1))
            u ← [Unpaired(U_1))]
            l ← l | Duplex(H, u)
            l ← l | Unpaired(U_2)
    s ← CutStruc(ŝ)
    return s
```

Algorithm A.2: The algorithm used to generate the structures in the engineered test set. The procedure EngineeredStructure(N) is called with $N \in \{50, 100, 200, 400\}$ and returns a structure of the corresponding length. This calls EngineeredTrial(N), which generates a candidate structure by iteratively sampling a loop, and appending a stem, hairpin, and unstructured region to the chosen loop. After reaching the target length, a nick is added and the candidate structure is returned. The candidate structure is checked to ensure that each hairpin is at least 3 nt long and that at least M_{cut} base pairs connect the strands. When these conditions are satisfied, the resulting target structure is returned.

```
BranchAndPropagate(v, 𝒟, 𝒞)
    i ← 1
    𝒟̄, V ← {}, {}
    𝒟̄_1 ← 𝒟
    while 1 ≤ i ≤ |v|
        𝒟 ← 𝒟̄_i
        v_k ← ChooseVariable(v, 𝒟)
        if v_k ≠ ∅
            V_i, 𝒟_k ← Instantiate(𝒟_k)
            success, 𝒟̂ ← Propagate(v, V, 𝒟, 𝒞)
            if success
                i ← i + 1
                𝒟̄_i ← 𝒟̂
            else
                𝒟̄_i ← 𝒟
        else
            𝒟̄_i, V_i ← ∅, ∅
            i ← i − 1
    return V
```

```
Propagate(v, V, 𝒟, 𝒞)
    success ← true
    changed ← true
    while changed and success
        changed ← false
        for 𝒞_i ∈ 𝒞
            for v_j ∈ v_{𝒞_i}
                for 𝒟_{j,k} ∈ 𝒟_j
                    if V ∪ {v_j ↦ 𝒟_{j,k}} ⇒ 𝒞_i
                        changed ← true
                        𝒟_j ← 𝒟_j \ 𝒟̃_j
    return V
```

Algorithm A.3: A depth-first branch and propagate algorithm that can be used to solve CSPs. The stack $\bar{\mathcal{D}}$ tracks the remaining domains after each instantiation. Specialized data structures and caching can reduce the cost beyond naive implementations. Research typically focuses on finding heuristics for choosing variables and instantiations, strengthening consistency to generate more implications, or breaking symmetries to reduce the search space.

```
HillClimbing(v, 𝒟, V^0)
    V ← V^0
    m^mutate ← 0
    C ← CalculateCost(V)
    while m^mutate < M^mutate
        μ, V̂ ← GenerateMutation(V, 𝒟)
        Ĉ ← CalculateCost(V̂)
        if Ĉ < C
            V ← V̂
        else
            m^mutate ← m^mutate + 1
    return V, C
```

```
IteratedLocalSearch(v, 𝒟)
    V ← Initialize(v, 𝒟)
    V, C ← HillClimbing(v, 𝒟, V)
    m^reopt ← 0
    while m^reopt < M^reopt
        μ, V̂ ← LargeMutation(V)
        V̂, Ĉ ← HillClimbing(v, 𝒟, V̂)
        if Ĉ < C
            V ← V̂
            m^reopt ← 0
        else
            m^reopt ← m^reopt + 1
    return V, C
```

Algorithm A.4: Iterated local search algorithm. Here we write the general form of an iterated local search algorithm. Compare to the OptimizeLeaves procedures in Algorithms 3.1 and 4.1.

Appendix B

Test tube design: supplementary information

B.1 Structural features of the engineered and random test sets

For the engineered test set, each dimer on-target structure was randomly generated, with stem and loop sizes randomly selected from a distribution of sizes representative of the nucleic acid engineering literature. For the random test set, each dimer on-target structure was generated by calculating the minimum free energy structure of a different random pair of RNA sequence at 37 °C. Target structures were selected so that the minimum cut (the number of intermolecular base pairs that must be broken to dissociate the dimer strands) was at least 9 bp for the engineered test set and at least 7 bp for the random test set. The structural properties of the on-target structures in the engineered and random test sets are summarized in Figure 3.5. Typically, the random test set contains on-target structures with a lower fraction of paired nucleotides (panel a), more stems (panel b), and shorter stems (panel c), and a higher minimum cut (panel d). The loop composition of each test set is summarized in Figure B.1. The target structures for the engineered test set, the random test set, and the *big tube test set* used for Figure 3.12 are provided as text files in the `test_tube` directory of the supplementary archive.

B.2 Selection and use of multiple exclusive split-points

During probability-guided decomposition, the set of exclusive split points $\{F\}$ is chosen to minimize the cost,

$$\text{cost}(\{F\}) = \sum_{F_i \in \{F\}} \left(|\phi_{k_{l_i}}|^3 + |\phi_{k_{r_i}}|^3 \right),$$

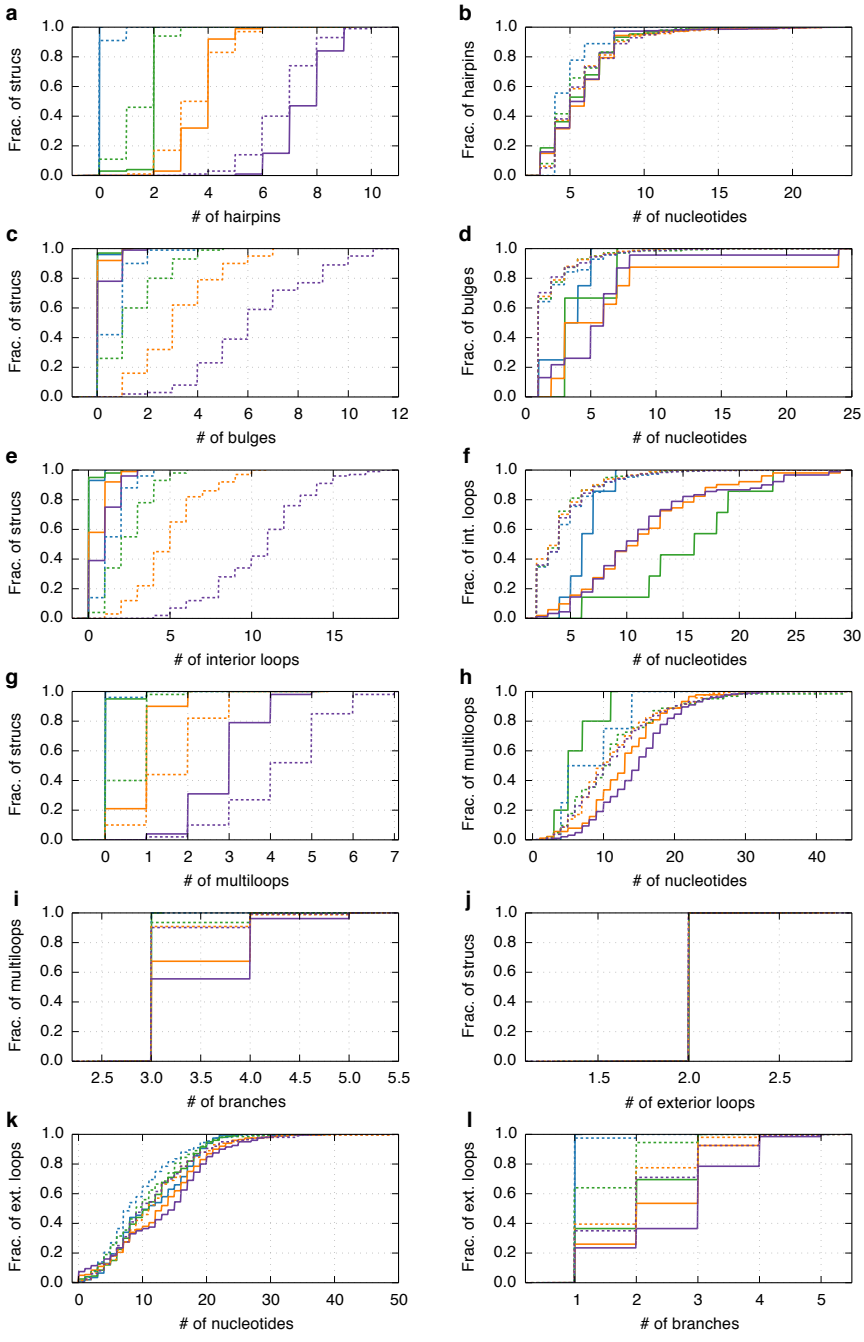


Figure B.1: Loop composition of the (dimer) on-target structures in the engineered (solid lines) and random (dashed lines) test sets. a) Number of hairpin loops per structure. b) Number of unpaired nucleotides in each hairpin loop. c) Number of bulge interior loops per structure. d) Number of unpaired nucleotides in each bulge loop. e) Number of non-bulge interior loops per structure. f) Number of unpaired nucleotides in each non-bulge interior loop. g) Number of multiloops per structure. h) Number of unpaired nucleotides in each multiloop. i) Number of branches in each multiloop. j) Number of exterior loops per structure (always 2 since these are all dimers). k) Number of unpaired nucleotides in each exterior loop. l) Number of branches in each exterior loop.

subject to constraints on the collective probability,

$$P(\{F\}) = \sum_{\{F\}} \min_{a \cdot b \in F^{\pm}} P_k^{a,b},$$

and exclusivity among split points (3.29). We use a depth-first branch and bound search procedure. At each branching step, a split point is added the current branch, $\{F\}$, greedily maximizing the collective probability while ensuring exclusivity among split points and sufficient leaf size among all potential children. The bound for each branch in the search tree is the cost of its split set. Upon backtracking, branches are explored only if their current bound is less than the minimum solution cost found so far, otherwise they are trimmed. If the probability constraints (3.29) are satisfied for an untrimmed branch, that set of split points is the current minimal cost solution and is therefore saved.

During structure- and probability-guided decomposition using multiple exclusive split-points, the first branch F is chosen from $B(S_k)$ (see (3.27)). Again, this branch is chosen to greedily maximize its probability. For structure-guided decomposition, the pair probability matrix P is set to be the structure matrix S of the target structure and the procedure is called as in structure- and probability-guided decomposition. The computational cost of this procedure is typically trivial compared to the cost of evaluating partition functions and pair probabilities.

Ensemble decomposition using multiple exclusive split-points allows ensemble properties to be estimated at low cost in some situations where single split-points do not. Figure B.2 shows the extent to which multiple exclusive split-points are used. From these graphs we see that only a small fraction of parents use multiple exclusive split-points (panel a), but almost 30% of complexes (panel b) and 80% of designs (panel c) include at least one parent with four children (and thus two split-points) for on-target lengths of 200 nt.

B.3 Algorithm performance for complex design

Here, we examine algorithm performance for the special case in which test tube design reduces to complex design: a target test tube containing one on-target complex and no off-target complexes. Figures B.3 and B.4 demonstrate that the performance of the current algorithm and the previously published single-complex design algorithm [83] is similar for the (dimer) on-target structures in the engineered and random test sets, respectively. Typical designs surpass the desired design quality (normalized ensemble defect ≤ 0.01; panel a), with the current algorithm overshooting the stop condition by a smaller margin. Typical design costs range from a fraction of a second for 50-nt on-target dimers in the engineered test set (B.3b) to 70 seconds for 400-ht on-target dimers in the random test set (B.4b). Starting from random initial sequences, the desired design quality can be

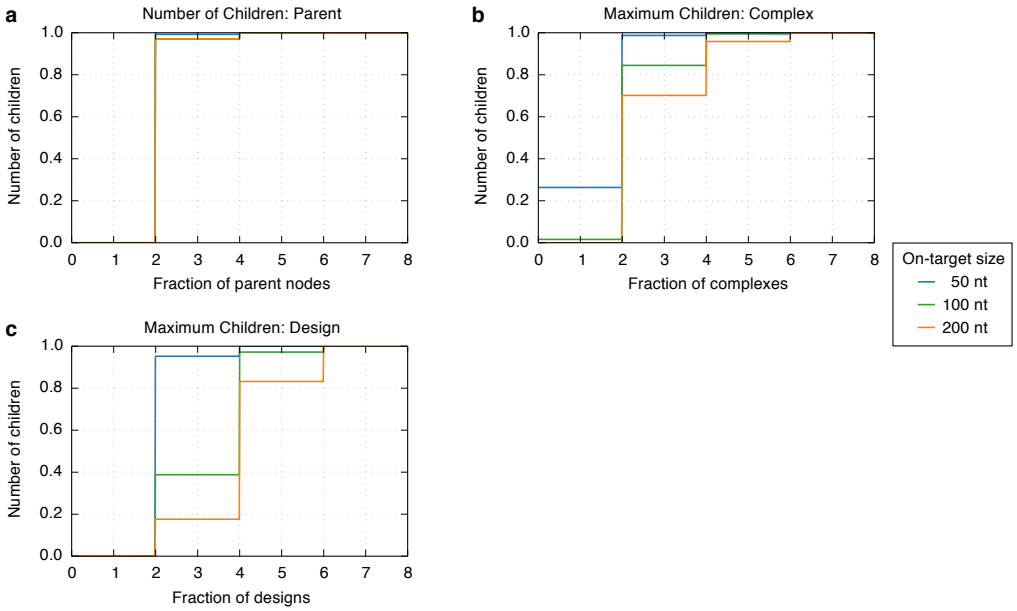


Figure B.2: Extent of multiple split point usage. a) Cumulative of the number of children per parent, b) cumulative histogram of the maximum number of children for any parent node in each complex, c) histogram of the maximum number of children for any parent node in each design. All data is for final decompositions for the engineered test set. RNA design at 37 °C.

achieved with a broad range of GC contents, with typical GC content less than 60% starting for both test sets. As the depth of the decomposition tree increases with increasing on-target size, the relative design cost, $\text{cost}_{\text{des}}/\text{cost}_{\text{eval}}$, decreases asymptotically towards the 4/3 optimality bound for typical design trials (panel B.3d).

B.3.1 Sequence initialization

Figure B.5 compares algorithm performance using different GC contents for random seeding and reseeding. Sequences were initialized with either random sequences (default), random sequences using only A and U, or random sequences using only G and C. The desired design quality is achieved independent of the initial conditions (panel a), with the typical design cost increasingly marginally if the initial sequence contains only G and C (panel b). Designs initiated with random AU or with random GC sequences illustrate that the desired design quality can be typically achieved over a broad range of GC contents (0.4 to 0.75). The typical cost of test tube design relative to a single evaluation of the test tube ensemble defect is within a factor of 4 (panel d).

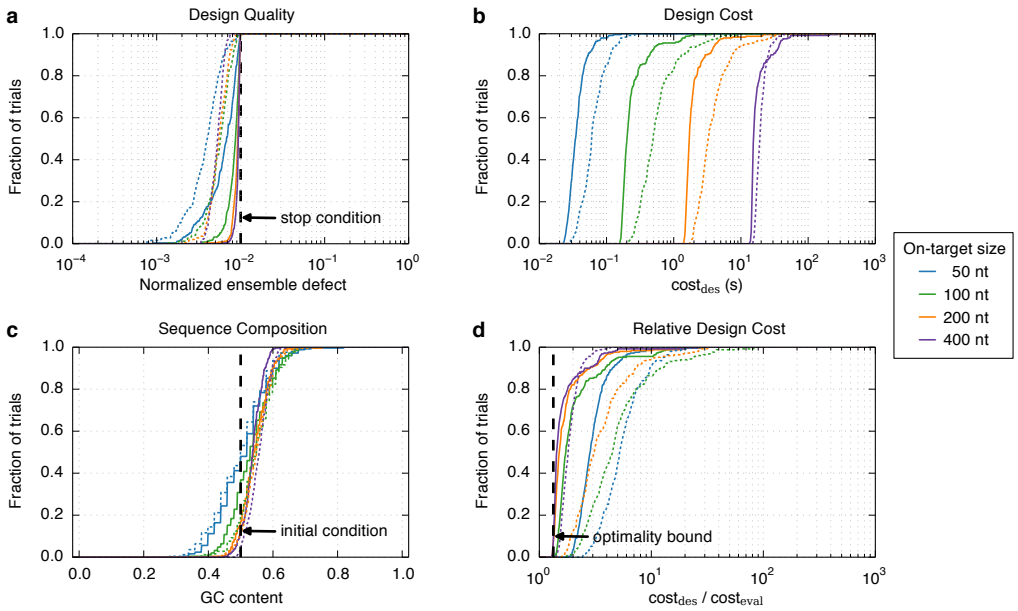


Figure B.3: Algorithm performance for complex design using the on-target structures from the engineered test set. Comparison of the current test tube design algorithm (solid lines) to the previously published single-complex design algorithm [83] (dashed lines). a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed line. d) Cost of sequence design relative to a single evaluation of the objective function. The optimality bound is depicted as a dashed line. RNA design at 37 °C. Each tube contains a single on-target dimer and no off-targets. There are 100 target tubes for each on-target size.

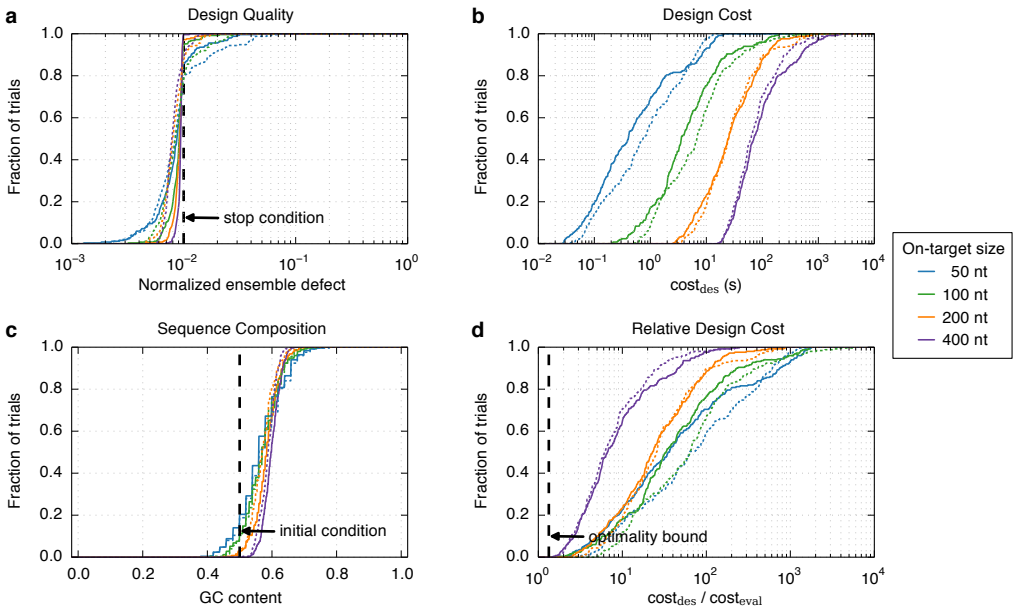


Figure B.4: Algorithm performance for complex design using the on-target structures from the random test set. Comparison of the current test tube design algorithm (solid lines) to the previously published single-complex design algorithm [83] (dashed lines). a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed line. d) Cost of sequence design relative to a single evaluation of the objective function. The optimality bound is depicted as a dashed line. RNA design at 37 °C. Each tube contains a single on-target dimer and no off-targets. There are 100 target tubes for each on-target size.

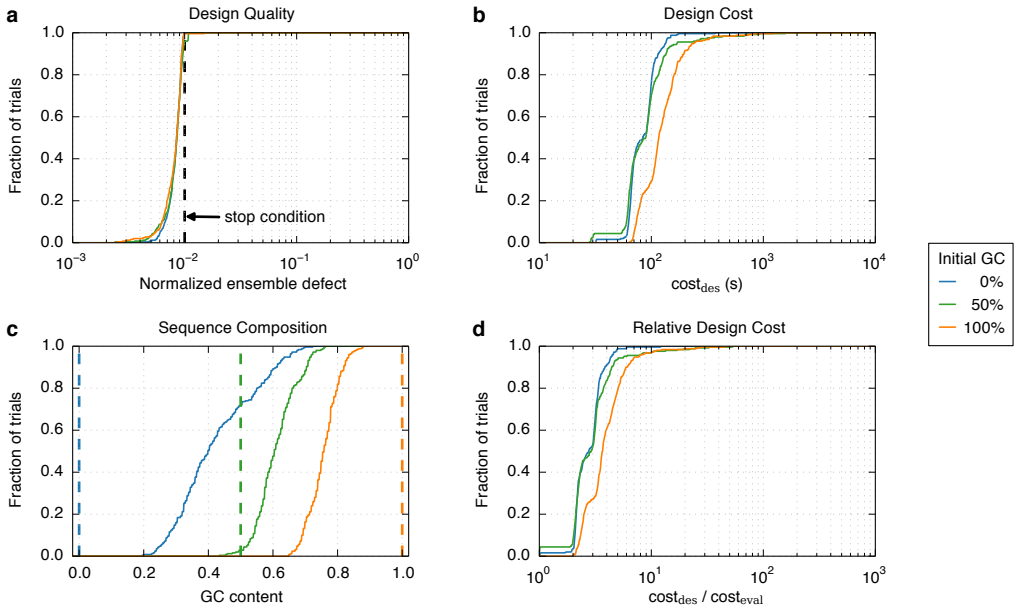


Figure B.5: Effect of sequence initialization on algorithm performance. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. c) Sequence composition. The GC contents used for seeding/reseeding are depicted as dashed lines. d) Cost of sequence design relative to a single evaluation of the objective function. RNA design at 37 °C for the subset of the engineered test set with 100-nt on-targets.

B.3.2 RNA vs DNA design

Figure B.6 compares RNA and DNA design. DNA designs are performed in 1 M Na^+ at 25 °C to reflect that DNA systems are typically engineered for room temperature studies. In comparison to RNA design, DNA design leads to similar design quality (panel a), marginally higher design cost (panels b and d), and comparable GC content (panel c).

B.4 Sensitivity of algorithm performance to design parameters

This section summarizes the results from sensitivity studies performed for each adjustable parameter in the paper, H_{split}, N_{split}, f_{split}, $f_{\text{stringent}}$, f_{redecomp}, f_{refocus}, f_{stop}, f_{passive}, M_{reopt}, and $M_{\text{unfavorable}}$. All studies were carried out with the 200 nt structures from the engineered test set. Two trials were carried out for each of the structures for a total of 100 trials per condition.

- Figure B.7 demonstrates the effect of changing the stop condition, f_{stop}. The 200-nt engineered and random test sets were designed with $f^{\text{stop}} \in \{0.001, 0.003, 0.01, 0.03, 0.1\}$. When the stop condition stringency is relaxed from 0.01, the defect correspondingly increases. When the stop

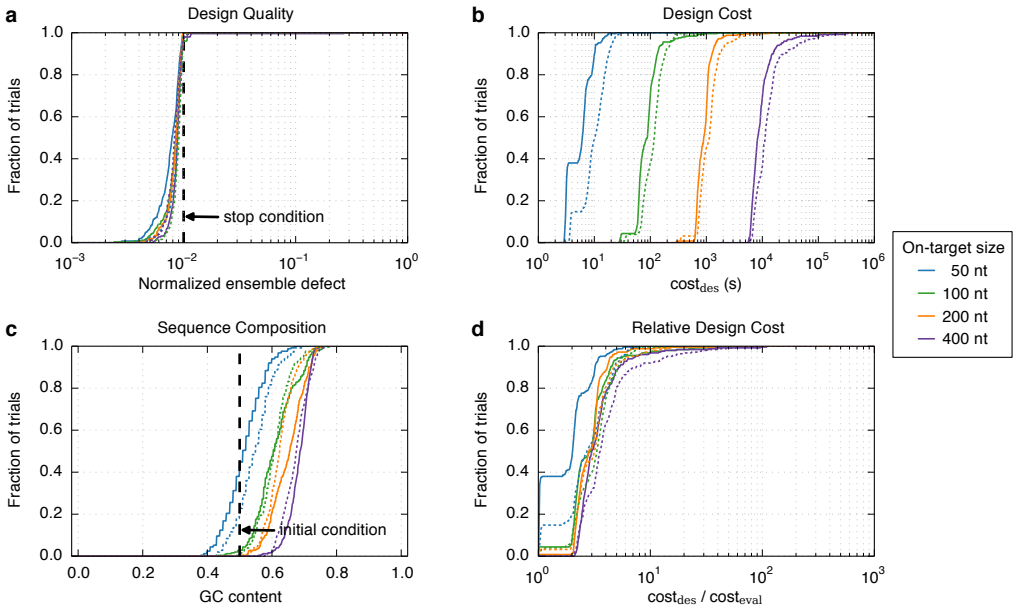


Figure B.6: Effect of design material on algorithm performance. a) Design quality. The stop condition is depicted as a dashed black line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed black line. d) Cost of sequence design relative to a single evaluation of the objective function. RNA design at 37 °C (solid lines) and DNA design at 25 °C (dashed lines) for the engineered test set.

stringency is increased ($f_{\text{stop}} \in \{0.001, 0.003\}$), the stop condition is typically not satisfied and design cost typically increases from 70 seconds to over 150 seconds. For a stop condition of 0.001, fewer than 20% of the resulting sequences satisfy their stop condition.

- Figure B.8 demonstrates the effect of changing $f_{\text{stringent}}$, the basal level of emergent defect allowed. We note little change in design quality or cost upon varying $f_{\text{stringent}}$ between 0.95 and 1.0. The value of $f_{\text{stringent}}$ only becomes apparent for designs which typically fail to satisfy their stop conditions. Figure B.9 demonstrates the effect of changing $f_{\text{stringent}}$ for the random test set when $f_{\text{stop}} = 0.03$. At $f_{\text{stringent}} = 1.0$, we see that 5-10% of design trials take the maximum allowed time. By not allowing any decomposition defect if the stop condition isn't satisfied, a small minority of design trials typically take much longer to complete.

- Figure B.10 demonstrates the effect of changing M_{reopt}, the number of consecutive leaf re-optimization attempts that must fail before leaf optimization terminates unsuccessfully. No substantial effect was seen on the resulting design quality or cost for the engineered test set, suggesting M_{reopt} is typically not exhausted on this test set. For the random test set, $M_{\text{reopt}} = 1$ results in designs failing to satisfy their stop condition more than 10% of the time.

- Figure B.11 demonstrates the effect of changing $M_{\text{unfavorable}}$, the number of consecutive leaf

mutation attempts that must fail before leaf mutation terminates unsuccessfully. The design quality and design cost were unchanged for the engineered test set except for trials where $M_{\text{unfavorable}} = 10$, when almost 20% of trials failed to satisfy their stop condition. The random test set shows a decreasing cost and an increasing fraction of satisfied trials as $M_{\text{unfavorable}}$ is increased.

- Figure B.12 demonstrates the effect of changing f_{redecomp}, the amount of defect that must be included on parental rejection. Design cost and quality was insensitive to this parameter for the 200 nt engineered test set. The design cost may be slightly reduced for $f_{\text{redecomp}} = 0.1$ for some trials in the random test set.

- Figure B.13 demonstrates the effect of changing f_{refocus}, the amount of defect that must be included on parental rejection and full test-tube rejection. For the engineered test set, the median design time decreases from 1000 seconds to 700 seconds as f_{refocus} decreases from 0.5 to 0.0. For the random test set, however, the median design time decreases until $f_{\text{refocus}} = 0$. At this point, the median design time increases by a factor of two.

- Figure B.14 demonstrates the effect of changing f_{passive}, the fraction of the allowed defect that is reserved for off-target complexes. For the engineered test set, little effect on the design cost or quality is noticeable for any non-zero value of f_{passive}. A zero value of f_{passive} increases the maximum observed design cost, but has little effect on the median cost. A similar effect is observed for the random test set. A value of $f_{\text{passive}} = 0.01$ appears to be superior than either of the other options.

- Figure B.15 demonstrates the effect of changing f_{split}, the probability accounted for by the exclusive split points. For both test sets, increasing f_{split} beyond 0.995 caused an increase in design cost and produced nearly equivalent design quality.

- Figure B.16 demonstrates the effect of changing H_{split}, the size of stable helix required at a split point and the length of the dummy nucleotide duplex. Varying H_{split} changes the ability of the algorithm to perform efficient decompositions and changes the quality of the root estimate. When $H_{\text{split}} = 1$, the resulting root estimate is sufficiently inaccurate that the design process typically fails to find high quality sequences. $H_{\text{split}} \geq 2$ designs typically satisfy their stop condition, overshooting by a smaller amount and taking longer as H_{split} is increased. Doubling H_{split} from 2 to 4 results in nearly an order of magnitude increase in the median design cost for the random test set.

- Figure B.17 demonstrates the effect of changing N_{split}, the minimum number of native nucleotides in a leaf. Varying N_{split} between 8 and 40 nucleotides shows a consistent increase in design cost as N_{split} is increased.

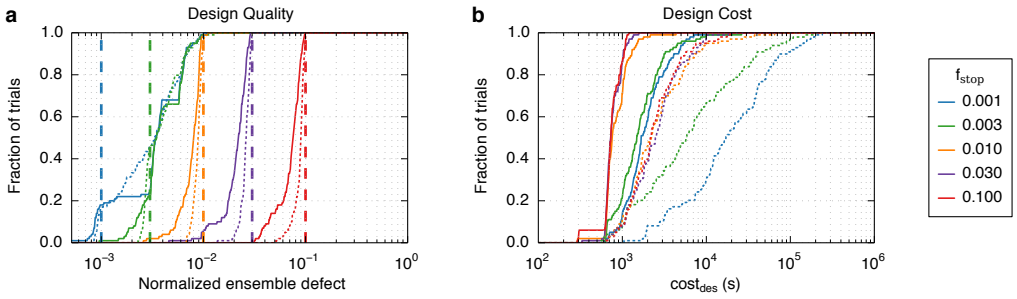


Figure B.7: Sensitivity to f_{stop}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

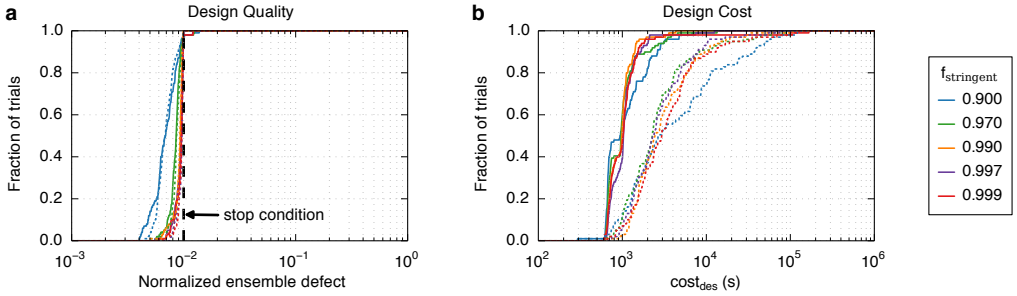


Figure B.8: Sensitivity to $f_{\text{stringent}}$. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

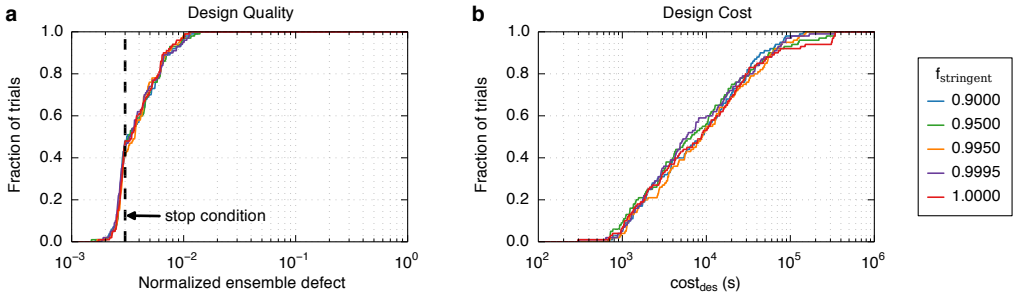


Figure B.9: Sensitivity to $f_{\text{stringent}}$ at $f_{\text{stop}} = 0.003$. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Random test set with $|s| = 200$, RNA design at 37 °C.

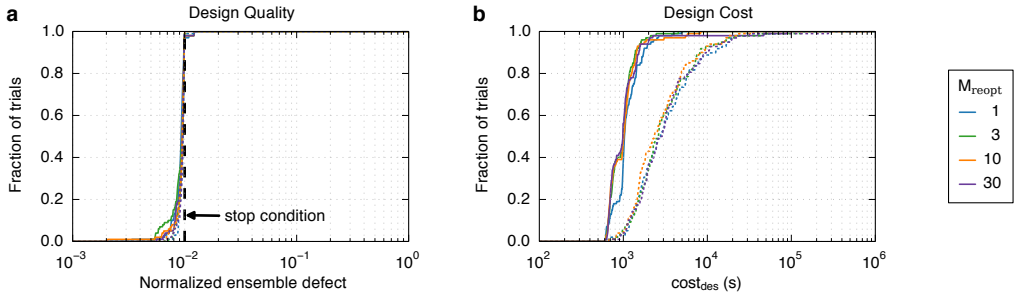


Figure B.10: Sensitivity to M_{reopt}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

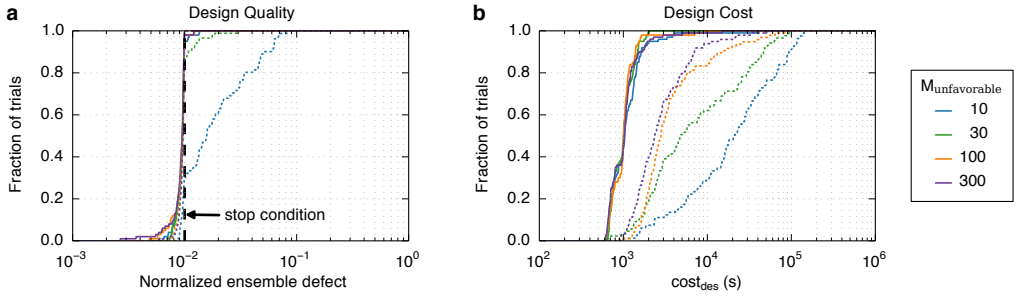


Figure B.11: Sensitivity to $M_{\text{unfavorable}}$. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

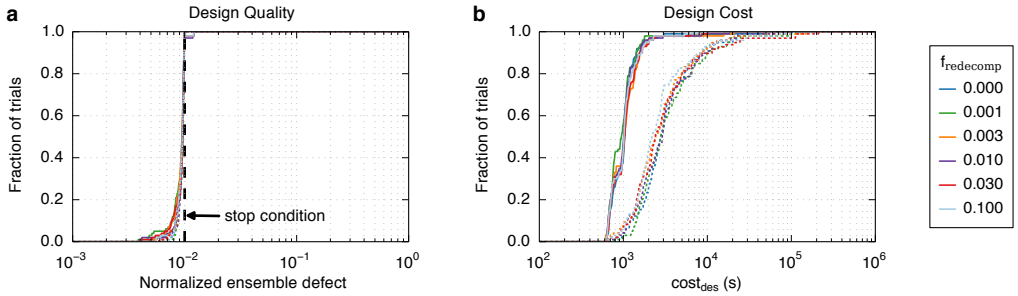


Figure B.12: Sensitivity to f_{redecomp}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

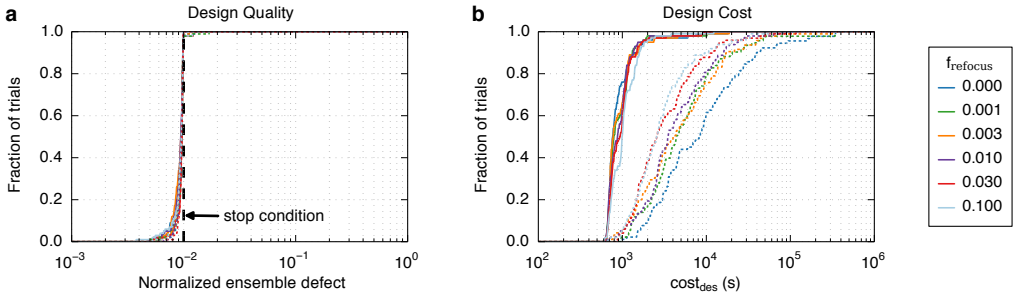


Figure B.13: Sensitivity to f_{refocus}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

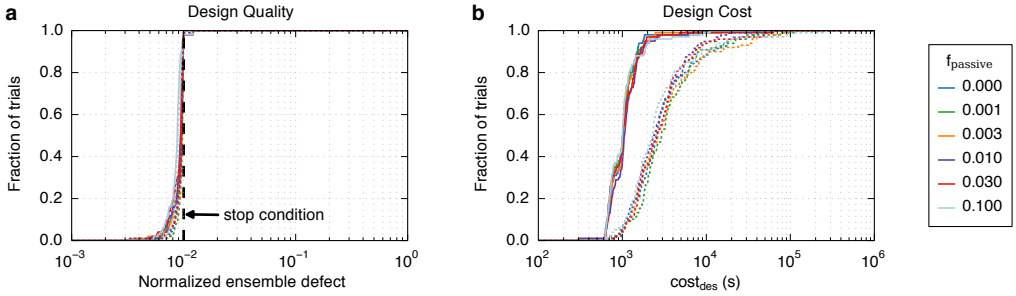


Figure B.14: Sensitivity to f_{passive}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

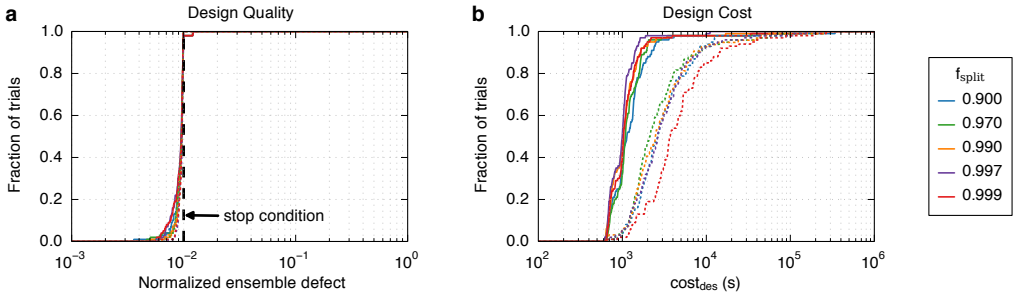


Figure B.15: Sensitivity to f_{split}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

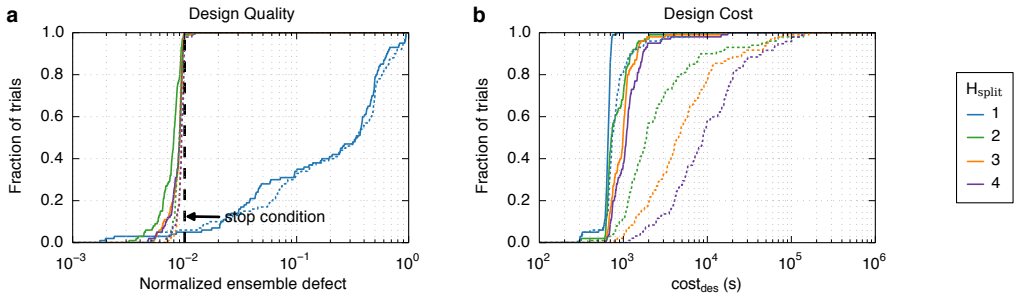


Figure B.16: Sensitivity to H_{split}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

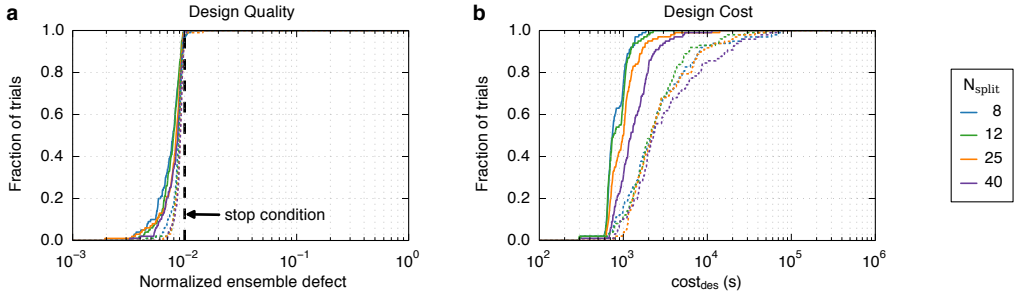


Figure B.17: Sensitivity to N_{split}. a) Design quality. The stop condition is depicted as a dashed line. b) Design cost. Engineered (solid lines) and random (dotted lines) test sets with $|s| = 200$, RNA design at 37 °C.

Appendix C

Nucleic acid reaction pathway design supplementary information

C.1 Supplementary results

C.1.1 Single complex and test tube designs: random test set

The multiobjective designer was tested on the random test set for single-complex design to ensure the performance matched the specialized single complex design algorithm. Figure C.1 shows the results of the single complex design algorithm, the test tube design algorithm, and the multiobjective design algorithm for the special case of single complex design on the random test set. The algorithms all perform similarly. Sequence quality (panel a), cost (panel b), GC content (panel c), and relative design cost (panel d) are nearly indistinguishable among the algorithms.

C.1.1.1 Tradeoffs between objective functions

In some cases, a nucleic acid engineer may wish to design a sequence that is compatible with multiple target structures. The multistate design algorithm allows specification of multiple target structures for a single complex, but stop conditions in this case need to be chosen carefully. To test the multistate algorithm on this design problem, two 50-nt target structures were chosen to have 16 nucleotides paired differently ($d(s_1, s_2) = 16$). Clearly, the lowest feasible sum of ensemble defects for any candidate sequence ϕ is

$$n(s_1, \phi) + n(s_2, \phi) \geq 16 \text{ nt}. \tag{C.1}$$

To optimize the structures to have a partitioning of the probability and to avoid overoptimizing one target structure at the expense of the other, we set $f_{\text{stop}} = 0.1$ for both target structures. The stop condition is infeasible, forcing optimization to continue until mutation and reoptimization attempts are exhausted.

The resulting sequence design and its design quality are shown in Figure C.2. The sequence

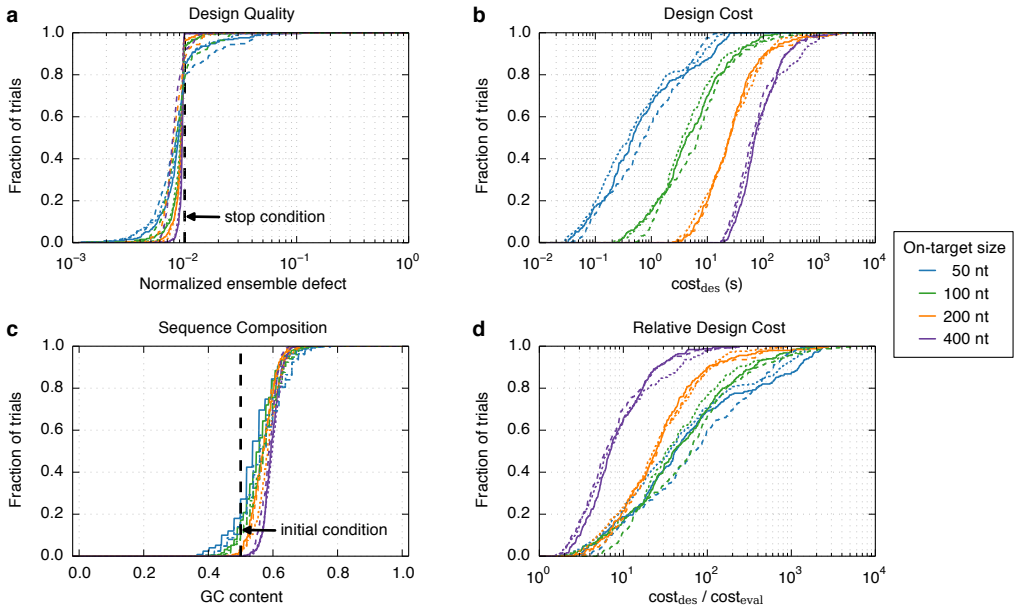


Figure C.1: Multistate algorithm performance for single complex design. Algorithm performance for test tube design. a) Design quality. The stop condition is depicted as a black dashed line. b) Design cost. c) Sequence composition. The initial GC content is depicted as a dashed line. d) Cost of sequence design relative to a single evaluation of the objective function. RNA design at 37 °C for the random test set for the Zadeh 2011 algorithm (dashed lines), the test tube design algorithm (dotted lines), and the multistate design algorithm (solid lines).

transitions one nucleotide from a Watson-Crick [G·C] in structure s_1 to a Wobble [G·U] in s_2 (panel a). The two target structures are close to isoenergetic for the designed sequence, and both ensemble defects are near 8 nt (panel b).

This result suggests that using stop conditions appropriately enables design of multistable sequences. Hierarchical decomposition using multiple exclusive split points allows estimation of the ensemble defect of all target structures for a complex using properties calculated inexpensively at the leaves of its decomposition tree.

C.2 Language definition

The scripting language is designed to be a descriptive domain specific language for defining nucleic acid reaction pathways. Here we specify the language using augmented Backus-Naur form (BNF).

For simplification, three aspects are not covered in the following BNF, comments, escaped newlines, and extra whitespace. Comments are Python-style comments, a comment starts with a # symbol and continues to the end of the line. If the last non-comment, non-whitespace character on a line is \, the next line is assumed to be a continuation of the current line. Whitespace is used only to separate tokens. Whitespace can intercede between any two tokens except within basic types, as

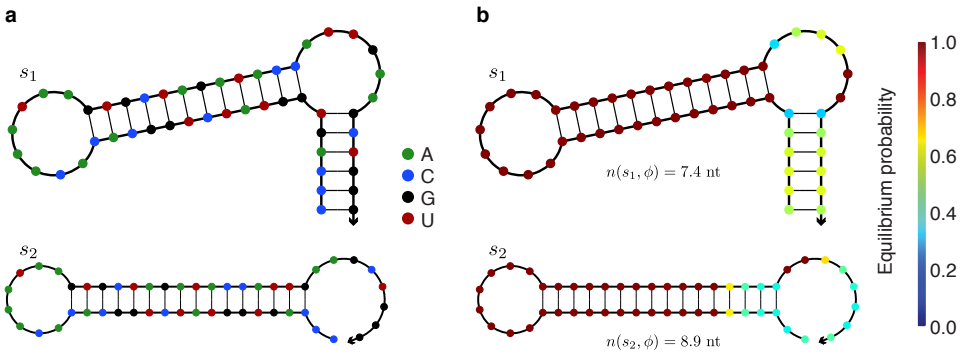


Figure C.2: Sequence designed with two target structures. a) The target structures with each nucleotide shaded according to its identity. b) The target structures with each nucleotide shaded according to its probability of being in the depicted state. The ensemble defect is labeled. ϕ = 5′–CCCAGUGGUAUCUGGCACACAAAUAAGUGCUAGAUACCAUUGAAGCUGGG–3′, RNA at 37 °C.

described below.

The starting symbol is ⟨input⟩ which contains zero or more statements. Each statement takes a single line

⟨input⟩ ⊨ ⟨statements⟩ | ϵ

⟨statements⟩ ⊨ ⟨statement⟩`\n` | ⟨statements⟩⟨statement⟩`\n`

⟨statement⟩ ⊨ ⟨structuredef⟩ | ⟨domaindef⟩ | ⟨tubedef⟩ | ⟨globaldef⟩ | ⟨dotdef⟩ |
⟨librarydef⟩ | ⟨dlistdef⟩ | ⟨sourcedef⟩ | ⟨namelistdef⟩ | ⟨matchdef⟩ | ϵ

The production for the basic types that follow must be contiguous (no whitespace).

⟨name⟩ ⊨ ⟨namestart⟩ | ⟨name⟩⟨namecont⟩

⟨namestart⟩ ⊨ _ | `A...Z` | `a...z`

⟨namecont⟩ ⊨ _ | `A...Z` | `a...z` | `0...9`

⟨integer⟩ ⊨ `0...9` | ⟨integer⟩`0...9`

⟨float⟩ ⊨ *A floating point number (1e-6, 1.0, etc)*

⟨domainname⟩ ⊨ ⟨name⟩ | ⟨name⟩`*`

⟨bool⟩ ⊨ `true` | `false`

C.2.1 Structure definitions

The following productions define a single target structure using DU-plus notation or dot-parens-plus notation.

⟨structuredef⟩ ⊨ `structure` ⟨name⟩`=`⟨structure⟩

⟨structure⟩ ⊨ ⟨dppstructure⟩ | ⟨dupstructure⟩

⟨dppstructure⟩ ⊨ ⟨dppstructure⟩`+`⟨onedppstructure⟩ | ⟨onedppstructure⟩

⟨onedppstructure⟩ ⊨ ⟨onedppstructure⟩`.` | ⟨onedppstructure⟩`(`⟨dppstructure⟩`)` | ϵ

⟨dupstructure⟩ ⊨ ⟨dupstructure⟩`+`⟨onedupstructure⟩ | ⟨onedupstructure⟩ | ϵ

⟨onedupstructure⟩ ⊨ ⟨onedupstructure⟩⟨dupelement⟩ | ϵ

⟨dupelement⟩ ⊨ ⟨dupunpaired⟩ | ⟨dupduplex⟩

⟨dupduplex⟩ ⊨ `D`⟨integer⟩⟨dupelement⟩ | `D`⟨integer⟩`(`⟨dppstructure⟩`)`

⟨dupunpaired⟩ ⊨ `U`⟨integer⟩

The following examples show the two ways of defining structures. The first, `S1`, is written in dot-parens-plus notation. The second, `S2`, is written in DU-plus notation.

```
structure S1 = ((((((((((......+))))))))))
structure S2 = D10 ( U6 + )
```

C.2.2 Sequence definitions

The following productions define domains, strands, and windows. Domains and strands are threaded onto target structures. Windows are used to define collections of domains that can be used in source-based constraints. Windows could be replaced with domains in future work.

⟨domaindef⟩ ⊨ `domain` ⟨domainname⟩`=`⟨sequence⟩

⟨sequence⟩ ⊨ ⟨sequenceelement⟩ | ⟨sequence⟩⟨sequenceelement⟩

⟨sequenceelement⟩ ⊨ ⟨nucleotide⟩ | ⟨nucleotide⟩⟨integer⟩

⟨nucleotide⟩ ⊨ `A` | `C` | `G` | `T` | `U` | `R` | `Y` | `M` | `K` | `W` | `S` | `B` | `D` | `H` | `V` | `N`

⟨domainlist⟩ ⊨ ⟨domainname⟩ | ⟨domainlist⟩ ⟨domainname⟩

⟨domainlistdef⟩ ⊨ ⟨domainlisttype⟩ ⟨domainname⟩`=`⟨domainlist⟩

⟨domainlisttype⟩ ⊨ `strand` | `window`

An example declaration of each type of element:

```
domain a = NNNNNNNNNN
domain b = N10
domain c = RRSSAUGCCA
domain d = R2 S2 AUGC2 A

strand s1 = a b* c
strand s2 = b* c* d
window w1 = a b*
```

C.2.3 Tube definitions

The following definition defines a tube to contain a set of target structures. The target concentrations are set using the `conc` properties defined later.

⟨tubedef⟩ ⊨ `tube` ⟨name⟩=⟨namelist⟩

An example tube declaration:

```
tube T = S1 S2
```

C.2.4 Advanced sequence constraints

Explicit external sequence constraints, complementary and identical domains, percent matches, and library definitions can be defined using the following productions.

⟨sourcedef⟩ ⊨ `source` ⟨name⟩=⟨sequence⟩

⟨namelistdef⟩ ⊨ ⟨relationtype⟩ ⟨domainlist⟩=⟨domainlist⟩

⟨relationtype⟩ ⊨ `complement` | `identical`

⟨matchdef⟩ ⊨ `match` ⟨name⟩=⟨sequence⟩

⟨librarydef⟩ ⊨ `library`⟨units⟩ ⟨name⟩=⟨seqlist⟩

⟨seqlist⟩ ⊨ ⟨sequence⟩ | ⟨seqlist⟩`,`⟨sequence⟩

Examples of each are shown below.

```
# A source sequence, note the backslashes at the end of the line
source GFP = \
```

```
auggugagcaagggcgaggagcuguucaccgggguggugcccauccuggu \
cgagcuggacggcgacguaaacggccacaaguucagcguguccggcgagg \
gcgagggcgaugccaccuacggcaagcugacccugaaguucaucugcacc \
accggcaagcugcccgugcccuggcccacccucgugaccacccugaccua \
cggcgugcagugcuucagccgcuaccccgaccacaugaagcagcacgacu \
ucuucaaguccgccaugcccgaaggcuacguccaggagcgcaccaucuuc \
uucaaggacgacggcaacuacaag

# specify a b c are equal to e f g
identical a b c = e f g

# specify the concatenation of a b c must be the reverse
# complement of x y z
complement a b c = x y z

# declare a match pattern
match q = AUGUCAGU

# declare a normal library
library select = CAGUGG, AGCUCG, CAGGGC

# declare an amino acid library. namespacing provided by the units
library[codons] I = ATT, ATC, ATA
library[codons] L = CTT, CTC, CTA, CTG, TTA, TTG
library[codons] V = GTT, GTC, GTA, GTG
library[codons] F = TTT, TTC
library[codons] M = ATG
library[codons] C = TGT, TGC
library[codons] A = GCT, GCC, GCA, GCG
library[codons] G = GGT, GGC, GGA, GGG
library[codons] P = CCT, CCC, CCA, CCG
library[codons] T = ACT, ACC, ACA, ACG
library[codons] S = TCT, TCC, TCA, TCG, AGT, AGC
library[codons] Y = TAT, TAC
library[codons] W = TGG
library[codons] Q = CAA, CAG
```

```
library[codons] N = AAT, AAC
library[codons] H = CAT, CAC
library[codons] E = GAA, GAG
library[codons] D = GAT, GAC
library[codons] K = AAA, AAG
library[codons] R = CGT, CGC, CGA, CGG, AGA, AGG
library[codons] STOP = TAA, TAG, TGA
```

The following properties are used to complete the definition of many constraints and objectives.

⟨units⟩ ⊨ `[`⟨name⟩`]` | `[%]` | ϵ

⟨namelist⟩ ⊨ ⟨name⟩ | ⟨namelist⟩ ⟨name⟩

⟨dnamelist⟩ ⊨ ⟨domainname⟩ | ⟨dnamelist⟩`.`⟨domainname⟩

⟨dotdef⟩ ⊨ ⟨dotnumerical⟩ | ⟨dotnamelist⟩ | ⟨dotrangelist⟩

⟨dotnumerical⟩ ⊨ ⟨dnamelist⟩⟨dotnumericalname⟩⟨units⟩`=`⟨float⟩

⟨dotnamelist⟩ ⊨ ⟨dnamelist⟩⟨dotnamelistname⟩⟨units⟩`=`⟨domainlist⟩

⟨dotrangelist⟩ ⊨ ⟨dnamelist⟩⟨dotrangelistname⟩⟨units⟩⟨nameorblank⟩`=`⟨rangelist⟩

⟨dotnumericalname⟩ ⊨ `conc` | `maxsize` | `stop` | `weight`

⟨dotnamelistname⟩ ⊨ `seq` | `libseq` | `source`

⟨dotrangelistname⟩ ⊨ `exclude` | `accessible` | `similarity` | `matchrange`

⟨rangelist⟩ ⊨ ⟨range⟩ | ⟨rangelist⟩⟨range⟩

⟨range⟩ ⊨ ⟨float⟩ | `[`⟨float⟩`,`⟨float⟩`]`

⟨nameorblank⟩ ⊨ ⟨name⟩ | ϵ

```
# Setting the sequence for a structure is done with the seq statement
# this sets the sequence to be strand s1 then strand s2
S1.seq = s1 s2

# Setting concentration for tube T containing structures S1 and S2
# Units are specified in brackets (allowed are M, mM, uM, nM, fM, aM)
T.S1.conc[uM] = 1

# Default units are molar
T.S2.conc = 1e-6
```

```
# The source for a window w is set to source sequence GFP with this
# constrains w to be a subsequence of GFP
w.source = GFP

# The sequence for domain a is constrained to be drawn from the concatenation
# of library sequences. Here we define the sequence of a based on the codon
# library.
a.libseq[codons] = I M C G

# domain a must match between 0 and 50% of the nucleotides of
# match pattern q.
q.matchrange[%] a = [0, 50]
```

C.2.5 Global parameters and options

Global properties, such as temperature, material, and N_{split} are set using the following productions.

⟨globaldef⟩ ⊨ ⟨globalnumerical⟩ | ⟨globalboolean⟩
| ⟨globalname⟩ | ⟨globalseqlist⟩

⟨globalnumerical⟩ ⊨ ⟨globalnumericalname⟩=⟨float⟩

⟨globalboolean⟩ ⊨ ⟨globalbooleanname⟩=⟨bool⟩

⟨globalname⟩ ⊨ ⟨globalnamename⟩=⟨name⟩

⟨globalseqlist⟩ ⊨ ⟨globalseqlistname⟩=⟨seqlist⟩

⟨globalnumericalname⟩ ⊨ `magnesium` | `sodium` | `seed` | `munfavorable` | `mleafopt`
| `fsplit` | `nsplit` | `hsplit` | `frelax` | `fpassive`
| `gcinit` | `minppair` | `ntrials` | `maxopttime`

⟨globalnamename⟩ ⊨ `material`

⟨globalnamename⟩ ⊨ `dangles`

⟨globalseqlistname⟩ ⊨ `prevent`

⟨globalbooleanname⟩ ⊨ `printsteps` | `printleaves`
| `allowwobble` | `disablemutweights`

Examples of global property settings:

```
temperature[C] = 23
temperature[K] = 310.15

magnesium[mM] = 100
sodium[M] = 0.9

seed = 1032

material = rna1999

dangles = all

prevent = AAAA, CCCC, GGGG, UUUU, GGCCUU
```

C.3 Example design scripts

Two types of design files are available in the supplementary archive (available through CaltechTHESIS). The first set of files are simple specifications intended to demonstrate features or design characteristics. A brief description of each specification is given in Section C.3.1.

The second set of designs are specifications generated from Python scripts. Two Python scripts are included in the archive file. The usage of each script and its output is described in Section C.3.2.

C.3.1 Simple examples

The following scripts are available as text files in the archive under `multistate/simple`. Each specification demonstrates a different type of design or constraint.

Single target structure. `00_singletarget.np` optimizes the complex ensemble defect of a single target structure.

Multiple target structures for a single sequence. `01_doubletarget.np` optimizes the complex ensemble defect of two target structures as described in Section C.1.1.1. Here, wobble pairing is enabled for implicitly defined constraints between base pairs in target structures.

Single test tube. `02_singletube.np` is an example from the random test set used during test tube design characterization. Note that no domains or strands are specified. If sequence constraints for a target structure are not specified, all strands in the target structure are assumed to be unconstrained except for implicit constraints defined by base pairs. The \ character is used to break structure definitions over multiple lines.

Multiple target structures. `03_multiplestruc.np` demonstrates how multiple target structures can be defined to optimize the structural ensemble of multiple complexes that share sequence information. Design of multiple complexes was first presented in Joseph Zadeh's thesis [81].

Multiple test tubes. `04_multipletube.np` specifies a design for a similar system using the test tube formulation presented in this thesis.

External sequence constraints. `05_externalseqs.np` specifies the design a reaction pathway using external sequence constraints. The sequences for two input strands are required to be subsequences of the given source sequences.

Library constraints. `06_codonlib.np` specifies the design of a reaction pathway using library constraints. The domain 'proteinseq' is constrained to match a list of codons.

C.3.2 Python-generated scripts

There are many possible ways to write design scripts. The test sets that were generated for this thesis and for several other design types in the Pierce Lab use Python scripts to automate the generation of large numbers of orthogonal systems. This method scales well to dozens of components. Within the `multistate/generated` directory of the thesis archive, there are two Python scripts, `make_multistate.py` and `make_hcr.py`. Each script is summarized below.

The script `make_multistate.py` was used to generate the test sets used in Chapter 4. When executed, it generates a directory for each design variant (with and without prevented patterns, etc.). Each subdirectory contains a single design type, labeled by the first author of the representative paper for the design type (`dirks`, `hochrein`, `seelig`, `yin`, and `zhang`). The directory below that contains a directory for each instantiation count, 1,..., 5. That directory contains 10 copies of each input file. In addition to the design specifications, a PBS queue file is saved in the `qsub_files` directory for each input file.

The script `make_hcr.py` was used to create DNA HCR design scripts. Sequences designed using a variant of this script and an early version of the multistate design algorithm were used in Choi et al. [11]. This script allows the user to specify the number of orthogonal systems, the length of the stem domain, b, and the length of the toeholds, a and c, via command line options. It saves the output into a file `hcr.np` in the current directory.

C.4 Extending the design algorithm

The design algorithm is written to be extended with additional constraints and design objectives. The algorithm implementation is in the `src/design/cpathway` directory of the NUPACK source distribution.

For examples on how to specify constraint types, inspect files `constraint_handler.h` and `constraint_handler.cc`. In particular, note the structure of the `propagate_constraint` functions. These functions determine infeasible instantiations of variables based on recently assigned variables.

For examples on how to specify design objectives, examine `objective_handler.cc` and `objective_handler.h`. These specify the standard test tube and complex ensemble defect objectives. To see how a combinatorial objective can be integrated, see the example sequence symmetry minimization implementation in `symmetry_calc.h`, `symmetry_calc.cc`, and `symmetry_objective.h`.

It may be beneficial to use an external library for constraint satisfaction instead of the current implementation. Any library would have to be released under a liberal open source license to be useful for the NUPACK source distribution. Only the constraint handler should need to be modified to integrate with a constraint satisfaction library.

Appendix D

Kinetic Coarse Graining: Supplementary Information

D.1 Large box stop condition

Unlike small box coarse-graining, satisfying an error bound on the total mass in unexplored reacting species does not guarantee an error bound on the accuracy of the simulations, even if all rates are known exactly. Consider a system that has one reaction,

$$A + C \rightleftharpoons B + C \tag{D.1}$$

where we start with $x_A = 1$ µM, $x_B = 0$ µM, and $x_C = 0.0001$ µM and $|\phi_A| = |\phi_B| = |\phi_C|$. In the current algorithm, molecule A would be chosen first, all unimolecular reactions and self-reactions would be explored, and none would be discovered. If $f_{\text{stop}} = 0.001$, the coarse-graining algorithm would then halt, regardless of $\tau_{\max}$, since there would be no flux into J^{u}. In reality, the concentration of B would gradually increase due to the catalytic effect of C. If the maximum estimated flux through reactions with unexplored species was used instead of just the flux through known reactions, it might be feasible to derive a more rigorous upper bound on the error that can be eliminated by further exploration.

D.2 Archive content

The archive file contains input files and scripts for the small and large box coarse-graining figures. The files are briefly described below along with references to the figures they were used in.

D.2.1 Small box input files

The files listed below were used in the small box section of the Chapter 5. All files are available in the archive in the `coarsegraining/small` directory.

- `threestate.in`: Used in Figure 5.3. This sequence has three nearly isoenergetic states and switches between them. It is useful for testing and exploring distance metrics at low cost. Figure 5.3 was generated with a slight modification of the normal coarse-graining algorithm that runs a final trajectory after identifying basins. During that trajectory the distances were printed to create the data for the plot.
- `exhaust.in`: Used in Figure 5.4. This input file was used to compare the coarse-graining to the full master equation solution.
- `othmer33.in`: This was used in Figure 5.5. It is also the same sequence as used in Figure 3.3 of Jonathan Othmer's thesis [51].
- `hcr.in`: This input file was used in Figure 5.6, the coarse-graining of HCR in the small box.

D.2.2 Large box input files

The files listed below were used in the large box section of the Chapter 5. All files are available in the archive in the `coarsegraining/large` directory.

- `andgate.in`: Used in Figure 5.8, the coarse graining of the AND gate [65].
- `cooperativegate.in`: Used in Figure 5.9, the coarse-graining of the cooperative gate [84].
- `hcr.in`: Used in Figure 5.10, the coarse-graining of HCR [11, 17].

D.2.3 Exhaustive enumeration

The exhaustively enumerated solution used in Figure 5.4 was simulated in Python. The script `kinutils.py` is available in the archive under `coarsegraining/exhaustive`.