Domain-Based Nucleic-Acid Minimum Free Energy: Algorithmic Hardness and Parameterized Bounds

Erik D. Demaine ✉
Massachusetts Institute of Technology, USA

Timothy Gomez ✉
Massachusetts Institute of Technology, USA

Elise Grizzell ✉
University of Texas Rio Grande Valley, USA

Markus Hecher ✉
Massachusetts Institute of Technology, USA

Jayson Lynch ✉
Massachusetts Institute of Technology, USA

Robert Schweller ✉
University of Texas Rio Grande Valley, USA

Ahmed Shalaby ✉
Hamilton Institute, Department of Computer Science, Maynooth University, Ireland

Damien Woods ✉
Hamilton Institute, Department of Computer Science, Maynooth University, Ireland

Abstract

Molecular programmers and nanostructure engineers use domain-level design to abstract away messy DNA/RNA sequence, chemical and geometric details. Such domain-level abstractions are enforced by sequence design principles and provide a key principle that allows scaling up of complex multistranded DNA/RNA programs and structures. Determining the most favoured secondary structure, or Minimum Free Energy (MFE), of a set of strands, is typically studied at the sequence level but has seen limited domain-level work. We analyse the computational complexity of MFE for multistranded systems in a simple setting were we allow only 1 or 2 domains per strand. On the one hand, with 2-domain strands, we find that the MFE decision problem is NP-complete, even without pseudoknots, and requires exponential time algorithms assuming SAT does. On the other hand, in the simplest case of 1-domain strands there are efficient MFE algorithms for various binding modes. However, even in this single-domain case, MFE is P-hard for promiscuous binding, where one domain may bind to multiple as experimentally used by Nikitin [Nat Chem., 2023], which in turn implies that strands consisting of a single domain efficiently implement arbitrary Boolean circuits.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Computer systems organization → Molecular computing

Keywords and phrases Domain-based DNA designs, minimum free energy, efficient algorithms, NP-hard, P-hard, NC, fixed-parameter tractable

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding Elise Grizzell is supported in part by the U.S. Department of Education grant P200A120256. Markus Hecher is supported by the Austrian Science Fund grant J4656 and the Society for Research Funding NOE grant ExzF-0004. Robert Schweller is supported in part by National Science Foundation Grant CCF2329918. Damien Woods & Ahmed Shalaby are supported by the European Research Council (ERC, Active-DNA, No. 772766); European Innovation Council (EIC, DISCO, No. 101115422); and Science Foundation Ireland (SFI) Nos. 18/ERCS/5746 & 20/FFP-P/8843.

Acknowledgements We thank Jenny Diomidova, Marco Rodriguez, Tim Wylie for helpful discussions.

1 Introduction

Computational prediction of nucleic acid systems plays a crucial role in their design, analysis, and engineering. For a system of DNA or RNA strands, we typically desire prediction of likely secondary structures—strand bindings formed by base pairing—at thermodynamic equilibrium, but ignoring 3D geometry, strain, kinetics, and many other details, as shown in Figure 1. The most favored secondary structure(s) at chemical equilibrium are those with *minimum free energy* (MFE). To assign a probability to any secondary structure at equilibrium, the *partition function*, the sum of the Boltzmann-weighted energy of each secondary structure, is used as a normalization factor. Typically, the space of secondary structures is exponential in system size, hence, efficient algorithms to compute them may or may not exist. Decades of work have produced beautiful connections between secondary structure features and algorithmic efficiency (see Section 1.2), as well as predictive software packages [13, 28] for system analysis and design. For molecular programming, showing that a class of systems is algorithmically hard to predict often implies they embed algorithms and, hence, might make good candidates for molecular computers.

1.1 Background and justification for domain-level analysis

Algorithms for thermodynamic secondary structure prediction research traditionally focus on the *base-level* of abstraction: strings over the alphabet A, C, G, and T for DNA, or U instead of T for RNA. However, DNA/RNA nanostructures and molecular programs are typically designed at a higher *domain-level* of abstraction, better suited to large systems with complicated interactions, which led Shalaby, Thachuk, and Woods [44] to propose seeking *domain-level thermodynamic algorithms* for predictive analysis. A domain d is a substrand of DNA/RNA that is assumed to bind perfectly to its complement domain d^*, and to no other (Figure 1), although variations of this definition are also used. The main motivations are twofold: (i) good DNA/RNA sequence design, and good system design principles, can be used to enforce a domain-based abstraction, and (ii) even with that simplified abstraction, the energy landscape is typically of exponential size; hence, the task of finding clever and efficient algorithms is still required for domain-level prediction. In general, multistranded and pseudoknotted systems either have no known efficient algorithms or are NP-hard to predict at the base (nucleotide) [1, 29, 30, 12] and/or domain [12] level. However, despite the lack of algorithmic thermodynamic prediction, multistranded and pseudoknotted domain-based nanostructure designs are some of the most successful to date, including DNA origami [40], RNA origami [22], and single/double-stranded tile systems [52, 54, 53, 18]. Clearly, the design process for these systems does not rely solely on full algorithmic prediction of secondary structure thermodynamics, but rather alternative methods, such as decomposing the system into smaller unpseudoknotted pieces [54, 18, 22] or by intuition-driven whiteboard sketches—all at the domain level. These successful experimental implementations give evidence for the benefits of domain-based design. Still, nevertheless, the lack of theoretical underpinning suggests a need for exclusively domain-based thermodynamic prediction algorithms [44] to continue along the journey of scale-up and complexification.

Since domains are merely a coarse-grained abstraction of DNA bases, the accuracy of domain-level models typically depends on good-quality DNA sequence design [54, 19, 48, 37, 55, 51], or on choosing biologically-sourced/random sequences with good enough properties [40]. Interestingly, domain-level design creates new challenges for thermodynamic prediction algorithms. Domain-level systems, like base-level systems, as noted above, tend to have exponentially large secondary structure spaces, meaning the existence of efficient

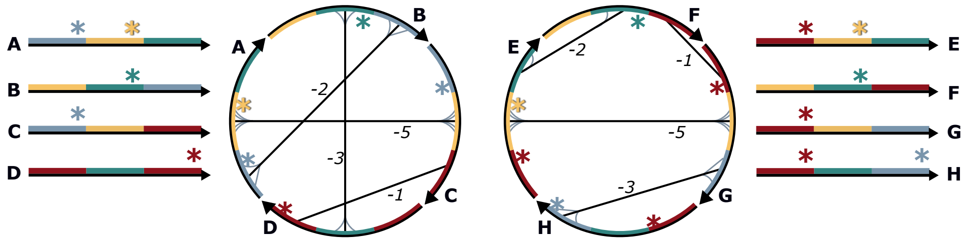


Figure 1 (Left): Domain-level system with 4 DNA strands having 3 domains each; codomains are indicated by $*$. (Middle-left): Example polymer graph of the strands showing domains bound a binding function δ with negative integer number strengths, e.g. $\delta(\text{green}, \text{green}^*) = -3$ (indicated both by numbers and by count of short grey/black curves attaching to a domain). (Right): Another domain-level system with four strands, having a maximally bound polymer graph with no crossings, showing that these strands have an unpseudoknotted MFE secondary structure.

(polynomial time) prediction algorithms may not be obvious. Further, domain-level systems may have an arbitrary number of domain types (base-level systems have only 4), as well as non-complementary (promiscuous) binding, meaning that the number of potential interactions in a system grows quickly with increasing system size.

1.2 Previous work on MFE and partition function

At the DNA base-level, for any n-strand connected secondary structure s, the free energy $\Delta G(s) = \sum_{l \in s} \Delta G(l) + (n-1)\Delta G^{\text{assoc}} + k_{\text{B}} T \log_{\text{e}} R$, where k_{B} is Boltzmann's constant in units of kcal/(mol $\cdot$ K) and T is temperature in Kelvin (K). In particular, this free energy includes the sum of the empirically-obtained free energies $\Delta G(l)$ of the constituent loops [13] of s, which are secondary structure features such as stack loops, hairpin loops, and others [41]. $\Delta G^{\text{assoc}} > 0$ is an entropic association penalty for bringing strands together, and there is an R-fold rotational symmetry penalty that is strictly positive for secondary structures with repeated strands that have several so-called rotational symmetries [13, 45]. The MFE of a set Ω of secondary structures is simply $\min_{s \in \Omega} \Delta G(s)$, and the partition function is the number $Q = \sum_{s \in \Omega} e^{-\Delta G(s)/k_{\text{B}} T}$. We use Q to define the probability of any secondary structure s at equilibrium: $p(s) = (e^{-\Delta G(s)/k_{\text{B}} T})/Q$.

At the domain-level, as in [44], we let $\Delta G(s) = \sum_{(d,e) \in \text{B}} \Delta G(d,e) + (n-1)\Delta G^{\text{assoc}}$, i.e. without any R-fold symmetry correction where B is the set of bonds of s and where $\Delta G(d,e)$ is the binding strength of domains d, e (defined more formally in Section 2).

Known algorithmic results

Almost all prior algorithmic work is at the DNA/RNA base-level, recent domain-level exceptions being [12, 44]. Single-stranded unpseudoknotted systems of length L bases have polynomial time $\mathcal{O}(L^4)$[1]. In 1990, McCaskill [32] showed dynamic programming efficiently calculates single-stranded partition function in polynomial time $\mathcal{O}(L^4)$, allowing computation of probabilities at equilibrium.

[1] We note that in the literature [13, 56, 12, 32] the polynomial is sometimes written to the power 3 (i.e. $\mathcal{O}(L^3)$ for single stranded and $\mathcal{O}(L^3(|\mathcal{S}|-1)!)$ for multistranded). This reduction in overhead comes from changing the standard energy model by putting some restrictions on the size of interior loops [13], or by enforcing certain mild conditions on the energy parameters for the interior loops [31, 25]

Multistranded systems. For systems with a constant number of strands ($|\mathcal{S}| = \mathcal{O}(1)$ strands, independent of total number of bases L), also unpseudoknotted, Dirks et al. [13], gave a polynomial time, $\mathcal{O}(L^4(|\mathcal{S}|-1)!)$, partition function algorithm, leaving MFE open. Recently, Shalaby and Woods [45] gave an $\mathcal{O}(L^4(|\mathcal{S}|-1)!)$ time algorithm for MFE in the same setting. In terms of computational complexity both of these problems are Fixed Parameter Tractable (FPT) with respect to strand count. For multi-stranded systems with a non-constant number of strands $|\mathcal{S}|$ and domain length L, Condon, Hajiaghayi, and Thachuk [12] showed a negative result: it is NP-complete [33] to predict MFE unpseudoknotted secondary structure(s), and even hard to approximate. They reduce from a variant of 3-dimensional matching (3DM) [20], with their result holding whether or not rotational symmetries are accounted for.
Pseudoknots. If we allow pseudoknots, there are as-of-yet unsolved modeling considerations: energy models are challenging to formulate due to the increased significance of geometric issues and tertiary interactions [13]. For simple energy models that allow pseudoknots, it is known that MFE prediction is NP-complete even for a single strand [1, 29, 30]. But, efficient dynamic programming algorithms exist for restricted classes of pseudoknots, for both MFE [39, 49, 11, 27, 38] and partition function [14, 15].
Domain-level. Two papers with domain-level algorithmic results are: Condon, Hajiaghayi, and Thachuk [12] showing multistranded MFE is NP-complete, and Shalaby, Thachuk, and Woods [44] giving a polynomial-time MFE algorithm for a subclass of multistranded systems—both papers utilize a long scaffold strand in different ways to give essentially opposite results.

1.3 Our Contributions

Our results, summarized in Table 1, mainly focus on MFE for multi-stranded systems with 1 or 2 domains per strand. Such few-domain systems are experimentally well-motivated: for example, SST systems [52] have only four domains per strand yet are capable of reasonably complicated computation [54], as are other tile systems [18, 43, 53, 4]. Nikitin [35] uses 1-domain promiscuous-binding to run depth-2 Boolean circuits, and there are strand displacement systems that compute using two [10] to a few [46, 55, 47] domains per strand.

We begin, in Section 2, with formal domain-based definitions of DNA secondary structures. In Section 3 we show there are small, 1 or 2 strand, systems with only 2 domains per strand that have pseudoknotted MFE structures (useful for later results). In our **first main result**, we show the simple-sounding case of 2-domain strands has NP-hard MFE (Theorem 14, Section 4). This uses the straightforward setting of perfectly complementary domains with all-equal binding strengths and improves the NP-hardness result of Condon, Hajiaghayi, and Thachuk [12], which required a long $\mathcal{O}(m)$-domain strand (for a 3DM instance with m triples). Both of these hardness results are then leveraged to give parameterized lower bounds on $|\mathcal{S}|$ and L, assuming the exponential time hypothesis (ETH) that there is no subexponential time algorithm for SAT.

We then investigate systems of strands with one domain (Section 5). Our **second main result**, Theorem 18, states that 1-domain systems, with promiscuous but bipartite binding and multiple strengths, are P-hard to predict (and hence likely unparallizable [33]). Moreover, this problem can be viewed as a natural generalization of the classic Edge Weighted Matching problem in which the vertex set is given as a multi-set with binary encoded counts. Showing that the Edge Weighted Matching problem is P-hard is a long-standing open problem [24]. Thus, the P-hardness of MFE for single-domain strands could provide important insights into this classic problem.

Theorem 19 gives an MFE algorithm running in time polynomial in the number of strands $|\mathcal{S}|$. Theorem 20 shows bipartite (domains and codomains) unit-strength binding is even easier, giving an $\mathcal{O}(|\Lambda|^3)$ time algorithm, i.e., an algorithm that is polynomial-time even when strand counts are provided in binary. Finally, for complementary binding, MFE is easier again as we have a sequential $\mathcal{O}(|\Lambda|)$ time one (Theorem 21), and a $\mathcal{O}(\log|\mathcal{S}|)$-time parallel algorithm (Theorem 22). The parallel algorithm puts this problem in the class NC [33], which taken together with Theorem 18 implies that promiscuous binding, multiset encoding, or both are needed for efficient simulation of sequential computation.

Our final result, Theorem 23, shows that the counting version of the free energy problem (that we call #FE) is #P-complete even for 1-domain strands and bipartite binding. While this doesn't show hardness for computing the partition function (PF), these problems are related since an efficient algorithm for #FE can be used to compute PF in $\mathrm{P}^{\#\mathrm{P}} = \mathrm{P}^{\mathrm{PP}}$ when the range of energy levels is polynomial. This relates PF to the counting hierarchy (CH) [50]. We also note that many of our results on 1-domain strands are reductions to or from the matching problem, the partition function of which, on regular graphs, has been investigated before [9, 6].

1.4 Future Work

For 1-domain strands, the main open question is to give an upper bound on the power of promiscuous binding with counts encoded in binary—shown here to be P-hard (Theorem 18). We believe this can be solved using b-matchings and thus P-complete[2]. Another interesting problem is whether the P-hardness result holds under further restrictions. If so, this must still take advantage of promiscuous binding or exponential strand count due to the NC result, Theorem 22.

What is the best run time for a FPT algorithm for MFE for strands with L domains which runs in time $2^{\mathcal{O}(|\mathcal{S}|)} \cdot L^{\mathcal{O}(1)}$ that accounts for rotational symmetry? We note that two recent papers give (a) an algorithm that handles rotational symmetry in the Turner/nearest neighbour model [45] (which could be ported to the domain model we use here, but with likely increase in run time due to the increase from 4 bases to $|\Lambda|$ domains), and (b) a singly-exponential algorithm that does not handle rotational symmetry [7] running in $O(3^{|\mathcal{S}|} \cdot L^3)$ time, making our lower bound tight up to ETH. The next interesting parameters to study are the number of domains $|\Lambda|$ or the number of strand types $|\Sigma|$.

2 Domain Based DNA Model

In this section, we discuss our DNA model and problems of interest.

▶ **Definition 1** (Domains, Codomain, and Strands). *A domain is a pair* (label, dir) *where* label *is a unique id usually represented by a letter and* dir $\in \{\rightarrow, \leftarrow\}$ *is a direction. The codomain of domain a is the domain with the same label and opposite direction, denoted by a^*. Let Λ be a set of domains, a strand $\sigma \in \Sigma$ is a sequence of domains all with the same direction (the strand is said to have that direction) denoted $\overrightarrow{ab}$ (for a 2-domain strand) and sometimes called $5'$ to $3'$ order. However, when it is clear that all domains have the same direction, we denote these as tuples (a, b). $\mathcal{S}$ denotes a multiset of strands, and $\Sigma = \mathrm{Supp}(\mathcal{S})$ denotes the support, or unique strand types, of $\mathcal{S}$.*

[2] This was pointed out after submission by Marco Rodriguez.

#Domains	Binding Type	Run Time Bounds	Complexity
L	Complementary & unit strength	UB: $\mathcal{O}(\lvert\Lambda\rvert L^3 \lvert\mathcal{S}\rvert^4 \cdot (\lvert\mathcal{S}\rvert-1)!)$ (**Thm. 15**), LBs: $2^{\Omega(\min(\lvert\mathcal{S}\rvert, L))}$ (**Thm. 17**)	NP-C [12]
2	Complementary & unit strength	UB: $\mathcal{O}(\lvert\Lambda\rvert \lvert\mathcal{S}\rvert^4 \cdot (\lvert\mathcal{S}\rvert-1)!)$ (**Thm. 15**), LB: $2^{\Omega(\lvert\mathcal{S}\rvert)}$ (**Thm. 16**)	NP-C (**Thm. 14**)
1	Promiscuous	UB: $\mathcal{O}(\lvert\mathcal{S}\rvert^4)$ (**Thm. 19**)	P† (**Thm. 19**), P-hard (**Thm.18**)
1	Bipartite unit strength	UB: $\mathcal{O}(\lvert\Lambda\rvert^3 \log \lvert\mathcal{S}\rvert)$ (**Thm. 20**)	P (**Thm. 20**)
1	Complementary	UB: $\mathcal{O}(\lvert\Lambda\rvert \log \lvert\mathcal{S}\rvert)$ (**Thm. 21**)	NC† (**Thm. 22**)

Table 1 Results for unpseudoknotted domain-level MFE. Upper bounds (UBs) are for a deterministic sequential algorithm with worst-case running time shown. All lower bounds (LBs) assume **ETH**. †Result holds for input **encoded in unary** and does not hold for input **encoded in binary.**

▶ **Definition 2** (Binding Function/Strength). *The binding function $\delta : \Lambda^2 \to \{0, -1, -2, \dots\}$ gives the binding strength between any two domains (more negative is more favorable).*

The previous definition assumes negative *integer* binding strengths between domains. We note that in the literature, more general rationals or reals (typically negative) are used for 'stack' energies [41], but our use of integers simplifies giving precise bounds on energy ranges.

We use the following definitions to classify the different types of binding functions:

- **Unit Strength:** For all $a, b \in \Lambda, \delta(a, b) \in \{0, -1\}$, i.e. non-0 binding strengths are equal.
- **Bipartite:** The domains can be partitioned into disjoint sets $\Lambda = \Lambda_D \bigcup \Lambda_C$, referred to as *domains* and *codomains*, such that for any two a, b in the same set $\delta(a, b) = 0$
- **Complementary:** We say a binding function is complementary if it is bipartite and there exists a perfect matching, meaning for all domains $a \in \Lambda_D$, there exists $a^* \in \Lambda_C$, such that $\delta(a, a^*) < 0$, and for all other pairs the binding strength is zero.
- **Promiscuous**: Any non-complementary binding function is said to be promiscuous (which may be bipartite or not).

▶ **Definition 3** (Domain-level strand system). *A domain-level strand system D, or simply system, is a multiset $\mathcal{S}$ of strands over support strand set $\Sigma = \mathrm{Supp}(\mathcal{S})$ and a binding function δ.*

▶ **Definition 4** (Domain-level secondary structure s). *For any domain-level strand system, a domain-level secondary structure, or simply secondary structure, s, is a set of domain pairs (hydrogen bonds, or simply bonds) respecting the binding function where no domain belongs to two pairs. Each domain is specified by a strand identifier and a position on that strand. For example, (i_p, j_q) denotes domain i of strand p binds to domain j of strand q such that $\delta(i_p, j_q) \neq 0$.*

Each secondary structure consists of one or more complexes:

▶ **Definition 5** (Complex). *A complex is a domain-pair connected domain-level secondary structure. Here, we also assume that each strand is connected: i.e. within each strand, consecutive domain pairs are connected (in their direction, i.e. $5'$ to $3'$ order).*

A ***polymer graph*** for a secondary structure s of a system D with multiset of strands Σ, and ordering of those strands π, is constructed by drawing them in π-order in the $5'$ to $3'$ direction around the circumference of a circle where: (i) the domains along each strand are assumed to be connected, in $5'$ to $3'$ order (by their *covalent bonds*), (ii) there is a ***nick*** (gap, i.e. no edge) between two adjacent strands and (iii) there is a chord connecting each domain pair (*hydrogen bond, or bond*) of s. Examples are shown in Figures 1–3. Let $|\mathcal{S}|$ denote the total number of strands (cardinality) in the multiset $\mathcal{S}$. The set of circular permutations, Π, of these $|\mathcal{S}|$ strands contains $(|\mathcal{S}|-1)!$ distinct circular permutations since cyclic permutations change the location of the strands on the circle without affecting their relative orderings (e.g., for three interacting strands $\{A, B, C\}$, $\Pi = \{ABC, ACB\}$ since the orderings ABC, BCA, and CAB are the same on a circle) [8]. Each circular permutation $\pi \in \Pi$ there has a distinct polymer graph.

▶ **Definition 6** (Pseudoknot-free, or unpseudoknotted, secondary structure)**.** *For any secondary structure s, we call s pseudoknot-free, or unpseudoknotted, if s has at least one circular permutation $\pi \in \Pi$ yielding a planar polymer graph (no crossing domain-pair edges), otherwise we call s pseudoknotted.*

In the following domain-based definition of free energy, we do not consider the entropic penalty due to rotational symmetry when there are repeated strands [13, 45].

▶ **Definition 7** (Free energy $\Delta G(s)$)**.** *The free energy, or simply energy, of a $|\mathcal{S}|$-strand, k-complex domain-level secondary structure s is $\Delta G(s) = \sum_{(a,b)\in s} \delta(a, b) + (|\mathcal{S}| - k)\Delta G^{\text{assoc}}$.*

▶ **Definition 8** (MFE secondary structure)**.** *For any domain-level strand system D, an MFE secondary structure is any unpseudoknotted secondary structure s such that $\Delta G(s) = \min_{s'\in\Omega} \Delta G(s')$, where Ω is the set of all unpseudoknotted secondary structures of D.*

An example of two polymer graphs, one pseudoknotted and the other unpseudoknotted, with their associated strands, can be found in Figure 1.

2.1 Problems and Parameterized Complexity

In computational complexity theory, it is useful to formalize problems as yes/no decision problems. In this paper, we are mainly concerned with the MFE decision problem, which asks whether the MFE of a system is below some threshold. This decision problem is in the class NP since one can give a secondary structure as a certificate and quickly, in polynomial time, compute its free energy and output yes/no depending on whether it is below the threshold.

▶ **Definition 9** (Minimum Free Energy (MFE) decision problem)**.** *Given a domain-level strand system and a number k, does there exist a secondary structure s such that $\Delta G(s) \leq k$?*

We assume the input to the MFE decision problem includes a **multiset** of strands (plus the binding function) where each strand is given as (σ_i, c_i) where $\sigma_i \in \Sigma$ is the strand type and c_i is an integer representing the number of copies of σ_i in the multiset $\mathcal{S}$. Due to this, we say an algorithm that runs in time $|\mathcal{S}|^{\mathcal{O}(1)}$ runs in **pseudopolynomial time**, since it runs in time polynomial in the cardinality of the multiset $\mathcal{S}$, i.e. the number of strands in the system, but not in the total input length (in bits). In some theorem statements, we refer to counts being **encoded in unary**, meaning the strands are given as a set with repeated strands written multiple times. This allows us to make claims about membership for "small" values. The goal of these statements is to show that hardness must make use of the multiset encoding, which has in other contexts been stated as Strong vs Weak NP-hardness [21].[3]

[3] A famous example of this is the partition problem [20] where we're given n integers and a value T and we want to know if there exists a subset of the number which sums to exactly T. This problem is solvable in time $\mathcal{O}(nT)$ but is NP-hard.

For our algorithms, we define our computational model to be **deterministic sequential RAM** machines with constant time memory access unless stated otherwise. We allow for constant time arithmetic of $\log_2 n$-bit numbers for input size n. This assumption does not speed up the run time of our algorithms by more than a factor of $\mathcal{O}(\log n)$.

We also consider the problem of counting the number of structures of free energy k.

▶ **Definition 10** (Counting Structures with Free Energy (#FE)). *Given a domain-level strand system and a value k, how many secondary structures s exist with $\Delta G(s) = k$?*

Fixed-Parameter Tractable (FPT) Algorithms run "fast" for instances with small parameters. For example, an algorithm that has a runtime of $f(k) \cdot n^{\mathcal{O}(1)}$ is said to be FPT in k. The **Exponential Time Hypothesis (ETH)** claims that there does not exist a $2^{o(n)}$ algorithm for SAT on n variables. This hypothesis establishes a technique for hardness and lower bounds by assuming ETH is true. By designing reductions that preserve parameters, we can achieve lower bounds for other problems such as MFE. These lower bounds are in the form of "There does not exist an algorithm that runs in time $2^{o(k)} \cdot n^{\mathcal{O}(1)}$".

3 Pseudoknots

Pseudoknots are surprisingly simple to form or avoid with short (few-domain) strands. We begin by establishing a condition for 2-domain strands that prevents the formation of pseudoknots. Then, we present short strands that have pseudoknotted minimum free energy (MFE) structures.

3.1 Pseudoknotted and Unpseudoknotted Systems

We define sided strands and show these cannot form pseudoknots. We use sided strands in the next section to avoid forming pseudoknots in our reductions. We show that this limit is somewhat tight in the sense relaxing this requirement allows for extremely simple pseudoknots to form in the domain-level model.

▶ **Definition 11** (Sided 2-domain strands). *A set of bipartite 2-domain strands is* sided *if every strand has the form (a, b^*) with $a \in \Lambda_D$ and $b^* \in \Lambda_C$*

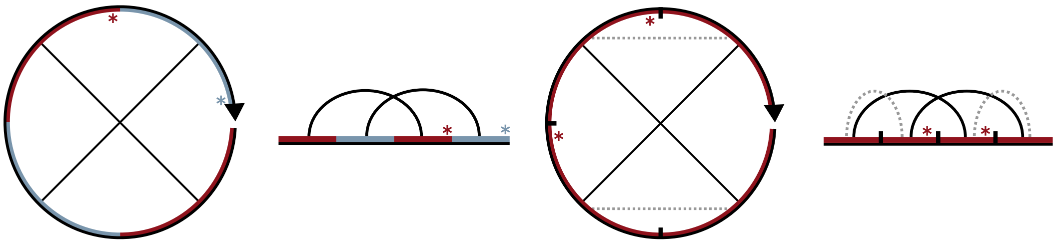
▶ **Theorem 12.** *Any secondary structure s containing only (≤ 2)-domain "sided" strands is unpseudoknotted, i.e. there is a strand order for s without crossings in the polymer graph.*

Proof. Recall that a secondary structure s includes a set of strands and their bonds. For any s, create an ordering on strands as follows. Select some sided strand (a, b^*) and add it to the drawing. If b^* is bound to another strand (b, c^*) in s then add that strand next in the ordering. Repeat this process until either (1) you reach a strand (d, a^*) where a^* is bound to a on the initial strand or (2) you reach a strand (d, z) where z is not bound to anything. Each adjacent strand added to the drawing has a bond drawn to its neighbor strand without crossing anything. If we end in case (1) we have built a cycle and can draw the new bond above all the others without crossing. If there still exist strands that are not yet added to the ordering, select one to add to the cycle then continue. ◀

Pseudoknots appear in MFE secondary structures, even for one or two strands:

▶ **Theorem 13.** *There exists a domain-level strand system with pseudoknotted MFE secondary structure s, with as few as 1 or 2 strands in the strand multiset $\mathcal{S}$. There are several scenarios:*

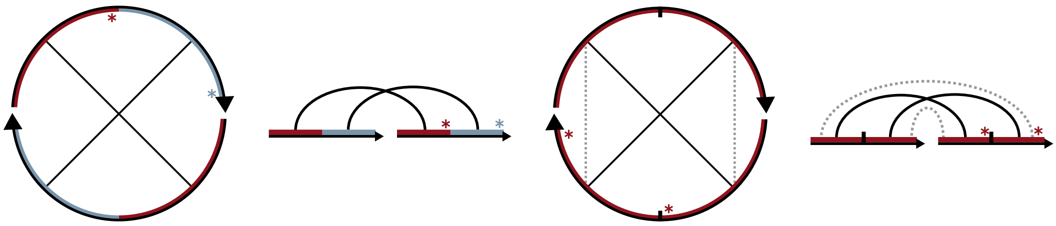
- $\mathcal{S} = \{((a, b, a^*, b^*), 1)\}$ *and s is the unique MFE secondary structure*



(a) Four-domain strand, containing 2 domain types and their complements, having pseudoknotted MFE secondary structure.

(b) Four-domain strand, containing 1 domain and its complement, having both pseudoknotted (solid curves) and unpseudoknotted (dotted curves) MFE secondary structures.

Figure 2 Single-stranded systems with only a few domains and pseudoknotted MFE structures.



(a) Two-domain strands, containing 2 domain types and their complements, having a pseudoknotted MFE secondary structure.

(b) Two-domain strands, containing 1 domain type and its complement, having both pseudoknotted (solid curves) and unpseudoknotted (dotted curves) MFE secondary structures.

Figure 3 Double-stranded systems with only a few domains and pseudoknotted MFE structures.

- $\mathcal{S} = \{((a, a^*, a^*, a), 1)\}$ *and s is not a unique MFE secondary structure*
- $\mathcal{S} = \{((a, b), 1), ((a^*, b^*), 1)\}$ *and s is the unique MFE secondary structure*
- $\mathcal{S} = \{((a, a), 1), ((a^*, a^*), 1)\}$ *and s is not a unique MFE secondary structure*

Proof. MFE secondary structures are in Figures 2 and 3. The single strand (a, b, a^*, b^*) is pseudoknotted since the only way to have two bonds is via a crossing. The strand (a, a^*, a, a^*) has two polymer graphs with 2 bonds, both with equal energy, although one has no crossings and the other does. The same proof can be seen for the cases with 2 strands. We note there is only one ordering for the case of 2 strands as the ordering is circular. ◀

4 Strands with 2 Domains

In this section, we show that the MFE problem is NP-hard even when strands contain only 2 domains. We show NP-hardness by reducing from the Directed 3-Cycle Cover problem with the promise that there are no doubly covered edges. This variant of 3-cycle cover asks: for a given graph, which does not have any 2-cycles, whether we can find a vertex-disjoint set of (directed) 3-cycles of the graph, such that all vertices are covered (occur in a 3-cycle). Theorem 24 in Appendix A shows this case of the cycle cover problem is NP-hard. We provide helper lemmas in the Appendix as well.

We start with the reduction from 3-cycle cover to the MFE problem in Theorem 14. This reduction holds even for unpseudoknotted structures from Theorem 12 as our strands are sided. Turning to parameterized complexity we describe the ETH based lower bounds, which follow from our reduction and the reduction from previous work [12]. We also cover the relation between these lower bounds and recent FPT algorithms shown in [45].

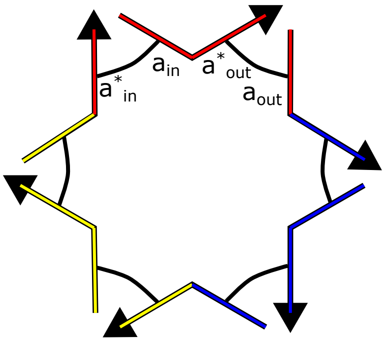


Figure 4 A complex representing a 3-cycle is built from 3 vertex strands and 3 edge strands. Each vertex strand is of a different color.

The Reduction

We now reduce the Directed 3-Cycle Cover with no 2-Cyles problem to MFE. Let $G = (V, E)$ be a given input directed graph with no 2-cycles. In this reduction, we create vertex strands and edge strands. Each cycle of our cover is represented by its own complex.

Domains. For each vertex $v \in V$ we create four domains , $v_{in}, v^*_{in}, v_{out}, v^*_{out}$. Domains marked with * are codomains in Λ_C. The binding function is complementary and unit strength.

Strands. For each vertex v we create a strand $\overrightarrow{v_{in}v^*_{out}}$. For each edge $(a, b) \in E$ we create a strand $\overrightarrow{a_{out}b^*_{in}}$

Complexes. The intuition behind this construction is that each complex will represent a valid path in G. Each secondary structure will then be a set of vertex disjoint paths. A cycle is represented by a complex that has no free domains. A 3-cycle is shown in Figure 4. We refer to the secondary structure in which all vertices are contained in (disjoint) 3-cycle complexes (if such a configuration exists) as a 3-cycle secondary structure.

▶ **Theorem 14.** *MFE of domain-level strand systems with 2-domain strands is NP-Complete. It remains NP-complete when restricted to sided strands, complementary binding, unit strength bonds, and single strand multiplicities.*

Proof. To show this, we provide an energy value k such that the MFE instance will have a secondary structure of energy less or equal to k if and only if the graph G has a 3-cycle cover, which implies NP-hardness by Theorem 24. Assume $0 < \Delta G^{\text{assoc}} < 1$ and set k as follows:

$$k = \Delta G(s) = -2n + \left(\frac{5n}{3}\right) \Delta G^{\text{assoc}}.$$

Now, suppose G has a 3-cycle cover, which implies the 3-cycle secondary structure exists. We know from Lemma 27 that this secondary structure s has $2n$ bonds and $(m - \frac{2n}{3})$ complexes, which implies s has energy:

$$\Delta G(s) = -2n + \left((n+m) - \left(m - \frac{2n}{3}\right)\right) \Delta G^{\text{assoc}} = -2n + \left(\frac{5n}{3}\right) \Delta G^{\text{assoc}} = k.$$

Therefore, such an MFE instance is a *yes* instance.

Now, suppose there is no 3-cycle cover. This means there is no 3-cycle secondary structure. By Lemma 29, we know the minimum energy secondary structure must have $2n$

bonds. Therefore, Lemma 28 implies that this structure must have fewer than $(m - \frac{2n}{3})$ complexes, which implies it has energy strictly greater than k, i.e. this MFE instance is a *no* instance. ◀

4.1 Parameterized Complexity

Beyond just hardness, we look at the MFE problem from a more fine-grained (parameterized) perspective. Precisely, we parameterize on the strand length L and number $|\mathcal{S}|$ of strands. We start by generalizing a known FPT algorithm [36] with respect to $|\mathcal{S}|$ to the domain-level model. Unfortunately, in general we can not avoid an exponential-time algorithm (unless the exponential time hypothesis fails) even for short strands $L \geq 2$. For fixed length 2 case we then give the conditional lower bound in Theorem 16 proven by our reduction. For the general case, we then give a combined lower bound in Theorem 17 based on the minimum of $|\mathcal{S}|$ and L. This shows the limits of FPT algorithms with respect to $|\mathcal{S}|$.

4.2 FPT Upper Bound

We prove this for bipartite unit-strength binding to compare against Theorem 14 and [12]. However these techniques should generalize incurring only a polynomial run time increase.

▶ **Theorem 15.** *MFE of domain-level strand systems with bipartite unit strength and pseudoknot free secondary structures is computed in time $\mathcal{O}(|\Lambda| L^3 |\mathcal{S}|^4 \cdot (|\mathcal{S}| - 1)!)$ for $|\mathcal{S}|$ strands of max length L over $|\Lambda|$ domain types.*

Proof. Consider a circular permutation π (out of $(|\mathcal{S}| - 1)!$ circular permutations) of the system strands. We use an extension algorithm of the single stranded maximum matching model algorithm [36]. The main extension is to include the multi-stranded case and the entropic penalties associated with it. The resulting recursion equation for the minimum free energy, $M_{i,j}$, of a subsequence Y of the ordering π, where Y runs the ith domain to j domain, is as follows:

$$M(i,j) = \min \begin{cases} M(i,k-1) + M(k+1,j-1) - 1 + \mathcal{I}(j,k)\Delta G^{\text{assoc}} \\ M(i,j-1) \end{cases}$$

Where $\mathcal{I}(j,k)$ is an indicator variable such that $\mathcal{I}(j,k) = 1$ iff both domains j and k belong to two different complexes. As we have two cases, (1) domain j does not form any domain-pair, or (2) domain j forms a domain-pair with some domain $k \in \{i, i+1, \ldots, j-1\}$. If domain j and k were belonging to two different complexes, then entropic penalty ΔG^{assoc} must be added, as they forming a domain-pair and hence reducing the number of complexes by one.

The algorithm will require a square matrix $M(i,j)$ as [36], but we augment each entry with a list of complexes that are formed in the minimum free energy structure within the the subsequence (i,j), such that each complex is a set of strands. Note that the size of all complexes at each entry can not exceed the number of strands. The value $\mathcal{I}(j,k)$ equals zero iff the two strands of domains j and k belong to the same complex (no entropic reduction) of the minimum free energy structure of the subsequence $(k+1, j-1)$ (found in the entry of $M(k+1, j-1)$), otherwise $\mathcal{I}(j,k)$ equals one (applying entropic penalty). We ensure choosing the appropriate k that guarantees that that domains $(j+1)$ and $(i-1)$ will be in the same complex if possible with augmenting this boolean value also (to ensure the least entropic penalty in future iterations), otherwise any k that minimize $M(i,j)$ works, which requires extra $\mathcal{O}(|\mathcal{S}|)$ time. Computing the augmented list of complexes at each entry follows directly based on k, and determining the value of $\mathcal{I}(j,k)$ takes then a constant time.

The time complexity of [36] is $\mathcal{O}(N^3)$ where N is the number of bases, in our case $N = \mathcal{O}(L|\mathcal{S}|)$ domains. So, the time complexity of our algorithm will be $\mathcal{O}(|\Lambda|L^3|\mathcal{S}|^4 \cdot (|\mathcal{S}|-1)!)$ considering the overhead of choosing the appropriate k that directly helps in determining the value of $\mathcal{I}(j,k)$, and the look up for the binding interaction of domains, and considering the whole possible circular permutations. ◀

Fixed Domain Length

For our reduction, we derive a lower bound of $2^{\Omega(|\mathcal{S}|)}$, even for $L = 2$, assuming ETH. This implies there does not exist a FPT algorithm with respect to strand length. In this case Theorem 15 gives a run time of $\mathcal{O}(|\Lambda||\mathcal{S}|^4 \cdot (|\mathcal{S}|-1)!) = \mathcal{O}(|\Lambda|) \cdot 2^{\mathcal{O}(|\mathcal{S}| \log |\mathcal{S}|)}$.

▶ **Theorem 16.** *MFE for domain-level strand systems with 2-domain strands requires time $2^{\Omega(|\mathcal{S}|)}$, unless ETH fails. This holds even when restricted to sided strands, complementary binding, and unit strength bonds.*

Proof. The result follows from Lemma 25 and Theorem 14. Observe that thereby $|\mathcal{S}|$ corresponds to $n + m$, where n is the number of variables and m is the number of edges. However, by using the so-called sparsification result [26] in advance, we can ensure both these terms are linear, giving the desired bound. Sparsification allows us to take advantage of the “trade-off" between the two parameters to achieve lower bounds on both n and m. ◀

Strand Count

For FPT algorithms we fix $|\mathcal{S}|$ to be some constant and consider $f(|\mathcal{S}|) \cdot \text{poly}(L, |\Lambda|)$ to be efficient. We are interested in getting a more precise estimate of $f(|\mathcal{S}|)$. Theorem 15 has a exponential factor of $O((|\mathcal{S}|-1)!)$. Without fixing strand length we show a lower bound of $2^{\Omega(|\mathcal{S}|)}$ (Theorem 17) using the 3DM reduction given by [12].

▶ **Theorem 17.** *MFE for domain-level strand systems with L-domain strands requires time $2^{\Omega(min(|\mathcal{S}|,L))}$, unless ETH fails.*

Proof. In the reduction by Condon, Hajiaghayi, and Thachuk [12], from 3DM, the long strand length m was equal to the number of sets in the 3DM proof. The number of strands in the system in $O(n)$. If we apply sparsification [26] first, we may assume that $m + n$ is linear in n. Consequently, the result directly follows from a $2^{\Omega(m)}$ ETH lower bound for 3DM [3]. ◀

5 Strands with 1 domain

In this section we first prove that MFE is P-hard for strands with only a single domain, and promiscuous binding (Theorem 18), by giving a simulation of Boolean circuits. The proof crucially uses multiple copies of each strand type. Then, we give three algorithms, the first of which (Theorem 19) shows that if the MFE problem is encoded in unary it is solvable in time $\mathcal{O}(|\mathcal{S}|^4)$, even with promiscuous binding. We then provide an algorithm for bipartite unit strength binding which runs in time $\mathcal{O}(|\Lambda|^3)$, Theorem 20. Our last algorithm shows easiness for complementary binding (Theorems 21 and 22).

5.1 P-hardness of single-domain promiscuous binding: Simulating Boolean circuits

In this section, we show P-hardness for MFE with single-domain strands by showing that computing MFE requires simulating/evaluating Boolean circuits. As shown later in the proof of Theorem 19, MFE with single-strand domains can be thought of as a weighted matching problem. Since weighted matching is not known to be P-hard [24], our P-hardness MFE result might be of independent interest as it shows P-hardness for a natural generalization of weighted matching in which the vertex set is given as a multi-set with binary encoded counts.

Some background on Boolean circuits and P-completeness. The circuit value problem (CVP) asks: given a Boolean circuit and its input, is the output 1? The problem is in P, as any circuit can be evaluated in time polynomial in circuit size and input length, and is P-hard since circuits efficiently simulate Turing machines, and that simulation (or reduction) can be encoded in one of the classes conjectured to be strictly in P (e.g. L, or NC^1; or with a little more work using a class known to be strictly contained in P, e.g. FAC^0). In 1977, Goldschlager [23] showed that monotone circuits, i.e. those that use only AND, OR, and input gates, are P-complete to predict. The trick is to use dual-rail logic: run one monotone circuit c on the input $x \in \{0,1\}^*$ and on its bitwise complement $\overline{x}$, run a 'complementary' monotone circuit denoted c' such that $\overline{c'(\overline{x})} = c(x)$. Since the dual-rail circuit is entirely monotone, a non-monotone reduction is used to convert x to $\overline{x}$. Even stronger, Theorem 6.2.5 of Greenlaw and Ruzzo [24], states that the following problem is P-complete: Synchronous, Alternating, Monotone Circuit-Value Problem with fanout exactly 2. Here, *synchronous* means that the circuit gates are organized into layers, where gates in layer i only take inputs from layer $i-1$. Every non-input gate has *fanout exactly 2*. Together with the property of being synchronous, this implies each non-input layer has the same number of gates (for decision problems, we only care about a single output bit of the circuit. Hence, there will be some redundant gates in the circuit). *Alternating* means that odd layers contain only OR gates and even layers only AND gates, except layer 0, which has input gates.

In a recent experimental paper, Nikitin [35] shows how to simulate 2 layer Boolean circuits by cleverly using what he terms "strand commutation" which is a form of promiscuous DNA strand binding using a mixture of mismatching and matching base pairs. Taking inspiration, we generalize his technique in several ways: (a) giving a proof that works for circuits of arbitrary depth, (b) having a fanin-2, fanout-2 gate design that has almost the same ΔG, except for multiples of some ϵ, for each of the 4 possible input bit pairs, (c) an overall circuit design for which the MFE is guaranteed to sit in an easily defined energy interval that is a simple function of circuit size and depth. Together, these properties are leveraged to establish the P-hardness of the MFE problem for single-domain systems. We note that this theorem holds in a generalization of the TBN model [16], where we allow promiscuous binding.

▶ **Theorem 18.** *MFE of domain-level strand systems with* 1*-domain strands, promiscuous (but bipartite) binding, and exponential strand counts, is P-hard to predict, under logspace reductions.*

Proof. Let C be any synchronous, alternating, monotone Boolean circuit where every gate has fanout exactly 2, and in particular, C uses only AND (fanin 2), OR (fanin 2), and input (fanin 0) gates. As discussed above, the problem of predicting families of such circuits is P-hard (Theorem 6.2.5 of [24]).

Let c be a copy of C and let c' be the dual circuit of c constructed as follows: For every non-input gate g in c, there is g' in c' where g' is OR iff g is AND, and vice-versa, and the input of c' is the bit-flipped input of c, with the wiring diagram being the same for both circuits (this is the standard dual-rail technique). Thus, for all gates g: $g(x_1, x_2) = \overline{g'(\overline{x_1}, \overline{x_2})}$ where $x_1, x_2 \in \{0,1\}$, and for the entire circuit $c(x) = \overline{c'(\overline{x})}$ where $\overline{x}$ denotes the bitwise complement of $x \in \{0,1\}^*$.

Simulating a single gate G. Intuitively, we wish to simulate each gate G in C using a strand gadget such that each gate in a layer has *almost* the same strand-gadget-MFE no matter which of the four input bit pairs G receives. Suppose C consists of a single gate G.

Suppose further that G is an AND gate. We claim that G is simulated by the 1-domain strand gadget in Figure 8 that operates by simultaneously simulating the corresponding AND (g) in c and OR (g') in c'. By simulate, we mean that (i) strands out_1 and out_2 are present in the MFE structure iff $g(x_1, x_2) = 1$, and (ii) that the gadget MFE lies in a real-valued interval to be defined later. Property (i) follows by a careful analysis of the binding energies $\delta_1 < \cdots < \delta_5$ (Figure 8), which are designed such that: input strands bind with strength δ_1 breaking up δ_2 bonds, freeing black strands to bind to the intermediate gadget green strands with δ_3 (or grey-green with $\delta_3 + \epsilon$, for some $\epsilon > 0$ to be defined later), with excess black strands binding to brown/orange with δ_4 to release pink outputs that were bound with δ_5.

If G is an OR gate, the same scheme is used except (a) all input bits and strands are flipped, and (b) the output comes from the OR component of Figure 8. Else, G is an input gate that is simulated by a single input strand type if $G = 1$ and zero strands if $G = 0$.

Gate at an arbitrary layer of an arbitrary C. Now let C be of arbitrary size. Let d be the depth of C and hence also of c and c' (we define the depth d to be the number of non-input layers), s be the size (including input gates), and $h = (s - |x|)/d$ be the height of C (or number of gates per non-input layer—since every gate has equal fanin and fanout of 2 (except for input gates), all non-input layers have the same number of gates h, and the input layer has $2h$ gates). The input layer is $\ell = 0$.

Let g, in layer $\ell > 0$, be any non-input gate in c, and let in_1 be any one of its 2 input wires and let out_1 be any one of its 2 output wires. The wire in_1 has an associated, unique strand type σ_{in_1}. The number of input strands (the count) of type σ_{in_1} is $|\sigma_{\text{in}_1}| = 2(2^{d-\ell})$ if the input bit is 1 and 0 if the input bit is 0. The number of output strands (the count) of type σ_{out_1} is $|\sigma_{\text{out}_1}| = 2^{d-\ell}$, if the output bit is 1 and 0 if the output is bit 0.

As shown in Figure 9, each gate has 11 strand types. We define gate g to have a total count of $26 \times 2^{d-\ell}$ strands, which can be seen as a multi-set of 11 strand types with repetition numbers shown in Figure 9.

We claim that the MFE, denoted by $k_g^{(a,b)}$, of any gate gadget g with any input bits (a, b), $a, b \in \{0,1\}$, has value in the negative integer range $[k_\ell, k_\ell + 2^{d-\ell+1}\epsilon]$ where $k_\ell = 2^{d-\ell}(4\delta_1 + 4\delta_2 + 2\delta_3 + 2\delta_4 + 2\delta_5)$. We will prove that claim by induction on $(d - \ell)$. For the base step, $(\ell = d)$, our construction in Figure 8 represents any final-layer gate $g_{\ell,i} = g_{d,i}$: specifically, the bottom of each of four Figure 8 panels shows that $k_{g_{d,i}}^{(a,b)}$ lies in the claimed interval, for each of the four cases of $(a, b) \in \{0,1\}^2$. Suppose that the claim is valid for any gate $g_{\ell,i}$ in layer ℓ such that $(d - \ell) > 0$ giving the following induction hypothesis: $k_{g_{\ell,i}}^{(a,b)} \in [k_\ell, k_\ell + 2^{d-\ell+1}\epsilon]$. Now, for any gate at the non-input layer $(\ell - 1)$, and from the recursive nature of our construction (Figure 10), leading to an exponential blow-up from right-to-left (towards the input), gives $k_{\ell-1} = 2k_\ell$, which implies that $k_{g_{\ell-1}}^{(a,b)} \in [k_{\ell-1}, k_{\ell-1} + 2^{d-\ell+2}\epsilon]$.

Let $E = \sum_{\ell \in \{1,2,\ldots,d\}} h 2^{d-\ell+1}\epsilon$ the sum of all ϵ's. Let $\epsilon = +1$. We will add an extra gadget, called the output gadget, which consists of a single strand that binds to the strand type out_1 of the single circuit output (final) gate, with binding strength $\delta_F = -E - 1$. Also, let δ_5^F be the δ_5 value for the circuit's final output gate: we set $\delta_5^F = \delta_F - 1$, and for each gate set $\delta_5 = \delta_4 + 1 = \delta_3 + 3 = \delta_2 + 4 = \delta_1 + 5$ to satisfy the inequality shown in Figure 8 and have integer-only strengths (a definition that propagates binding strengths back through circuit gadgets, from output back to inputs).

Formula for MFE. We next claim that c (and thus C) accepts iff MFE $< \sum_{\ell \in \{1,2,\ldots,d\}} k_\ell h$. To see this note that *without the output gadget* (i.e. ignoring δ_F) the MFE is in the negative integer interval:

$$\left[\sum_{\ell \in \{1,2,\ldots,d\}} k_\ell h \ , \ E + \sum_{\ell \in \{1,2,\ldots,d\}} h k_\ell \right] \tag{1}$$

but since $\delta_F < -E$, we know that the MFE including the output gadget (i.e. including δ_F) lies below the interval in (1) and this will happen iff circuit C accepts its input x.

We claim the reduction is computable by deterministic logarithmic space Turing machine [34, 2] that takes input C, x: We assume the circuit is described in a standard way as a string [5]. The circuit height h and depth d are easily computed in logspace (e.g. count the number of gates that take input from the first layer to give h, and divide that into circuit size to get d). Each gate description includes 11 strand types, unique to the gate (Figures 8 and 9), which are straightforward functions of the gate name. For each strand type, its count is a function of circuit depth and gate layer (Figure 9) that uses multiplication and exponentiation, on binary numbers of $O(|x|^{O(1)})$ bits (these numbers are powers of 2 so could be written using $O(\log |x|)$ bits, although that is not required here since logspace machines can output polynomial-sized words). Likewise for the MFE threshold value: $\sum_{\ell \in \{1,2,\ldots,d\}} k_\ell h$. The binding function (Figure 8), for any pair of strands, is a simple formula of the depth. All gates at layer l have the same binding function as they do not interact with each other. Hence, at layer l, the binding strength $\delta_1 = -(E + 1 + 6d)$, and $\delta_2, \ldots, \delta_5$ values follow directly as described above. This value of δ_1 guarantees that $\delta_F = -E - 1$ (E is a power of 2, so all δ's could be written using $O(\log |x|)$ bits). ◀

5.2 Polynomial-time algorithms for simulating 1-domain systems

▶ **Theorem 19.** *MFE of domain-level strand systems with 1-domain strands is solvable in $\mathcal{O}(|\mathcal{S}|^4)$, even for promiscuous binding functions. With unary encoded counts, this problem is in P.*

Proof. Create a graph G where each node is a strand. Multiple strands of the same type have multiple nodes. For every pair of nodes representing strands with domains a and b, add an edge with weight $(-1)\delta(a,b) - \Delta G^{\text{assoc}}$ (to make weights positive).

Each weight is then the contribution of a complex to the energy. Since we are computing a matching, each strand will be used once. The graph size will be $|\mathcal{S}|$ and the upper bound on the number of edges is $|\mathcal{S}|^2$. Since MAX weight matching has a $\mathcal{O}(V^2 E)$ time algorithm, this gives a $\mathcal{O}(|\mathcal{S}|^4)$ algorithm for MFE. ◀

Bipartite Unit Strength

▶ **Theorem 20.** *MFE of domain-level strand systems with 1-domain strands, bipartite binding, and unit-strength bonds, is in P and solvable in $O(|\Lambda|^3 \log |\mathcal{S}|)$, even for promiscuous binding functions.*

Proof. We solve this by reducing it to the max-flow problem. Let $A = a_1, a_2, \ldots, a_n$ and $B = b_1, b_2, \ldots, b_m$ denote the bipartite partition for the domains of a given MFE instance, and let $c(x)$ denote the strand count for a given domain x (i.e. the number of strands with domain x). Create a network flow instance as follows: create a network with a source s, sink t, and a vertex for each domain type $a_1, \ldots, a_n, b_1, \ldots, b_m$. Connect the source s of the network to each a_i with a capacity $c(a_i)$ edge, and connect each b_j to t with a capacity $c(b_j)$ edge. Add an edge of capacity ∞ from a_i to b_j if $\delta(a_i, b_j)$ is non-zero (i.e. if a_i bonds to b_j). The max-flow of this network is equal to the maximum possible bonds achievable by any configuration for the given MFE input and, therefore, can be used to determine the solution to MFE in polynomial time $\mathcal{O}(|\Lambda|^3)$. We add an additional $\log|\mathcal{S}|$ factor to this run time to account for arithmetic based on strand counts. ◀

Complementary Binding. Lastly, we show that Theorem 18 *requires promiscuous binding* single-domain strands. First, we describe how this problem can be solved sequentially in time $\mathcal{O}(|\Lambda|)$, then describe how to parallelize it:

▶ **Theorem 21.** *MFE of domain-level strand systems with* 1*-domain strands and complementary binding is solvable in time* $\mathcal{O}(|\Lambda|\log|\mathcal{S}|)$.

Proof. Each strand with domain a can only bond with its codomain a^*. This means the number of complexes for that domain type pair is the smaller of the two numbers, which we write as $\min(|a|, |a^*|)$. We can then compute the binding strength times number of complexes $\delta(a, a^*)\min(|a|, |a^*|)$ to get the first term of the function $\Delta G(s)$ for an MFE secondary structure s. The number of removed complexes is also $\min(|a|, |a^*|)$, which we can multiply by ΔG^{assoc} to get the contribution of the second term. In total, we are making $|\Lambda|$ comparisons, each of two numbers $\leq |\mathcal{S}|$. Then we are summing up $|\Lambda|$ minima, and returning it. We add a $\log|\mathcal{S}|$ factor to the run time to account for the cost of arithmetic operations. ◀

The next result shows that the algorithm from Theorem 21 can be parallelized to get an NC algorithm. Hence, MFE of single-domain, complementary binding systems cannot be P-hard unless NC=P, in turn implying that non-complementary binding, i.e. promiscuous, is likely required for efficient (polynomial time) simulation of arbitrary sequential computations (Theorem 18).

▶ **Theorem 22.** *MFE of domain-level strand systems with* 1*-domain strands and complementary binding is in NC when encoded in unary.*

Proof. For NC membership, we require, at most, polylogarithmic time on a polynomial number of processors. The algorithm from Theorem 21 can be parallelized by computing the smaller value between domains and codomains on $|\Lambda|$ different processors. This can be done in $\mathcal{O}(\log|\mathcal{S}|)$ time. Then, we add the free energy contributions of each domain pair in parallel, taking $\mathcal{O}(\log|\Lambda|\log|\mathcal{S}|)$ parallel time in total. ◀

5.3 Counting Free Energy

The counting problem #FE is still hard, which we establish below. We show that there exists a parsimonious reduction from counting matchings to #FE.

▶ **Theorem 23.** *Counting the number of structures with energy* E *is #P-Complete even with bipartite unit strength binding and encoded in unary.*

Proof. We reduce from Bipartite Matching. For each vertex, we create a domain v. For each edge, we make the binding strength of both domains equal to -1. The set of configurations of bonds is equivalent to the sets of edges. The energy of each configuration is a function of the number of edges represented. Thus, if we can compute the number of configurations with energy level E in polynomial time, we then determine the number of matchings. ◀

References

1. Tatsuya Akutsu. Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Applied Mathematics*, 104(1-3):45–62, 2000.
2. Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
3. Nikhil Bansal, Tim Oosterwijk, Tjark Vredeveld, and Ruben Van Der Zwaan. Approximating vector scheduling: almost matching upper and lower bounds. *Algorithmica*, 76:1077–1096, 2016.
4. Robert D Barish, Rebecca Schulman, Paul WK Rothemund, and Erik Winfree. An information-bearing seed for nucleating algorithmic self-assembly. *Proceedings of the National Academy of Sciences*, 106(15):6054–6059, 2009.
5. David A Mix Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41(3):274–306, 1990.
6. Alexander Barvinok and Alex Samorodnitsky. Computing the partition function for perfect matchings in a hypergraph. *Combinatorics, Probability and Computing*, 20(6):815–835, 2011.
7. Kimon Boehmer, Sarah J Berkemer, Sebastian Will, and Yann Ponty. Rna triplet repeats: Improved algorithms for structure prediction and interactions. 2024. 24th Workshop on Algorithms in Bioinformatics (WABI), to appear.
8. Richard A Brualdi. *Introductory combinatorics*. Pearson Education India, 1977.
9. Andrei Bulatov and Martin Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2):148–186, 2005. Automata, Languages and Programming: Algorithms and Complexity (ICALP-A 2004).
10. Luca Cardelli. Two-domain DNA strand displacement. *Mathematical Structures in Computer Science*, 23(2):247–271, 2013.
11. Ho-Lin Chen, Anne Condon, and Hosna Jabbari. An $O(n^5)$ algorithm for MFE prediction of kissing hairpins and 4-chains in nucleic acids. *Journal of Computational Biology*, 16(6):803–815, 2009.
12. Anne Condon, Monir Hajiaghayi, and Chris Thachuk. Predicting Minimum Free Energy Structures of Multi-Stranded Nucleic Acid Complexes Is APX-Hard. In Matthew R. Lakin and Petr Šulc, editors, *27th International Conference on DNA Computing and Molecular Programming (DNA 27)*, volume 205 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:21, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.
13. 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):65–88, 2007.
14. Robert M Dirks and Niles A Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of computational chemistry*, 24(13):1664–1677, 2003.
15. Robert M Dirks and Niles A Pierce. An algorithm for computing nucleic acid base-pairing probabilities including pseudoknots. *Journal of computational chemistry*, 25(10):1295–1304, 2004.
16. David Doty, Trent A Rogers, David Soloveichik, Chris Thachuk, and Damien Woods. Thermodynamic binding networks. In *DNA23: The 23rd International Conference on DNA Computing and Molecular Programming*, volume 10467 of *LNCS*, pages 249–266. Springer, 2017.
17. Martin E. Dyer and Alan M. Frieze. Planar 3DM is NP-Complete. *J. Algorithms*, 7(2):174–184, 1986.
18. Constantine G. Evans. *Crystals that count! Physical principles and experimental investigations of DNA tile self-assembly*. PhD thesis, Caltech, 2014.
19. Constantine G Evans, Jackson O'Brien, Erik Winfree, and Arvind Murugan. Pattern recognition in the nucleation kinetics of non-equilibrium self-assembly. *Nature*, 625(7995):500–507, 2024.
20. M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

21 Michael R Garey and David S Johnson. “strong” NP-completeness results: Motivation, examples, and implications. *Journal of the ACM (JACM)*, 25(3):499–508, 1978.

22 Cody Geary, Paul WK Rothemund, and Ebbe S Andersen. A single-stranded architecture for cotranscriptional folding of RNA nanostructures. *Science*, 345(6198):799–804, 2014.

23 Leslie M Goldschlager. The monotone and planar circuit value problems are LOG SPACE complete for P. *ACM SIGACT news*, 9(2):25–29, 1977.

24 Raymond Greenlaw, H James Hoover, and Walter L Ruzzo. *Limits to parallel computation: P-completeness theory*. Oxford University Press, USA, 1995.

25 Ivo L Hofacker, Christian M Reidys, and Peter F Stadler. Symmetric circular matchings and RNA folding. *Discrete mathematics*, 312(1):100–112, 2012.

26 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

27 Hosna Jabbari, Ian Wark, Carlo Montemagno, and Sebastian Will. Knotty: efficient and accurate prediction of complex RNA pseudoknot structures. *Bioinformatics*, 34(22):3849–3856, 2018.

28 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*, 6:1–14, 2011.

29 Rune B Lyngsø and Christian NS Pedersen. Pseudoknots in RNA secondary structures. In *Proceedings of the fourth annual international conference on Computational molecular biology*, pages 201–209, 2000.

30 Rune B Lyngsø and Christian NS Pedersen. RNA pseudoknot prediction in energy-based models. *Journal of computational biology*, 7(3-4):409–427, 2000.

31 Rune B Lyngsø, Michael Zuker, and CN Pedersen. Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics (Oxford, England)*, 15(6):440–445, 1999.

32 John S McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers: Original Research on Biomolecules*, 29(6-7):1105–1119, 1990.

33 Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, 2011.

34 Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, 2011.

35 Maxim P Nikitin. Non-complementary strand commutation as a fundamental alternative for information processing by DNA and gene regulation. *Nature Chemistry*, 15(1):70–82, 2023.

36 Ruth Nussinov and Ann B Jacobson. Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proceedings of the National Academy of Sciences*, 77(11):6309–6313, 1980.

37 Lulu Qian and Erik Winfree. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034):1196–1201, 2011.

38 Jens Reeder and Robert Giegerich. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC bioinformatics*, 5:1–12, 2004.

39 Elena Rivas and Sean R Eddy. A dynamic programming algorithm for RNA structure prediction including pseudoknots. *Journal of molecular biology*, 285(5):2053–2068, 1999.

40 Paul WK Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, 2006.

41 John SantaLucia Jr and Donald Hicks. The thermodynamics of DNA structural motifs. *Annu. Rev. Biophys. Biomol. Struct.*, 33:415–440, 2004.

42 Thomas J. Schaefer. The Complexity of Satisfiability Problems. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 216–226. ACM, 1978.

43 Rebecca Schulman and Erik Winfree. Synthesis of crystals with a programmable kinetic barrier to nucleation. *Proceedings of the National Academy of Sciences*, 104(39):15236–15241, 2007.

44 Ahmed Shalaby, Chris Thachuk, and Damien Woods. Minimum free energy, partition function and kinetics simulation algorithms for a multistranded scaffolded DNA computer. In Ho-Lin Chen and Constantine G. Evans, editors, *29th International Conference on DNA Computing and Molecular Programming (DNA 29)*, volume 276 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1:1–1:22, Dagstuhl, Germany, 2023. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

45 Ahmed Shalaby and Damien Woods. An efficient algorithm to compute the minimum free energy of interacting nucleic acid strands, 2024. Arxiv preprint: `arXiv:2407.09676`.

46 Friedrich C Simmel, Bernard Yurke, and Hari R Singh. Principles and applications of nucleic acid strand displacement reactions. *Chemical reviews*, 119(10):6326–6369, 2019.

47 David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, 2010.

48 Niranjan Srinivas, James Parkin, Georg Seelig, Erik Winfree, and David Soloveichik. Enzyme-free nucleic acid dynamical systems. *Science*, 358(6369):eaal2052, 2017.

49 Yasuo Uemura, Aki Hasegawa, Satoshi Kobayashi, and Takashi Yokomori. Tree adjoining grammars for RNA structure prediction. *Theoretical computer science*, 210(2):277–303, 1999.

50 Klaus W Wagner. The complexity of combinatorial problems with succinct input representation. *Acta informatica*, 23:325–356, 1986.

51 Boya Wang, Chris Thachuk, Andrew D Ellington, Erik Winfree, and David Soloveichik. Effective design principles for leakless strand displacement systems. *Proceedings of the National Academy of Sciences*, 115(52):E12182–E12191, 2018.

52 Bryan Wei, Mingjie Dai, and Peng Yin. Complex shapes self-assembled from single-stranded DNA tiles. *Nature*, 485(7400):623–626, 2012.

53 Erik Winfree, Furong Liu, Lisa A Wenzler, and Nadrian C Seeman. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394(6693):539–544, 1998.

54 Damien Woods, David Doty, Cameron Myhrvold, Joy Hui, Felix Zhou, Peng Yin, and Erik Winfree. Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. *Nature*, 567(7748):366–372, 2019.

55 David Yu Zhang and Georg Seelig. Dynamic DNA nanotechnology using strand-displacement reactions. *Nature chemistry*, 3(2):103–113, 2011.

56 Michael Zuker and Patrick Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic acids research*, 9(1):133–148, 1981.

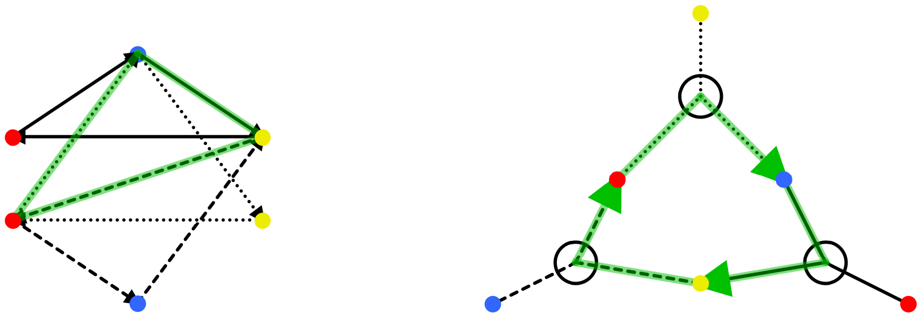


Figure 5 (Left): Interleaving of three different (solid, dotted, dashed) directed 3-cycles could cause a new 3-cycle that is not among the given ones. (Right): Such a 3-cycle required a non-subdivided, triangular face formed from all three colors of the 3-cycles, not occurring in the reduction (Figure 7).

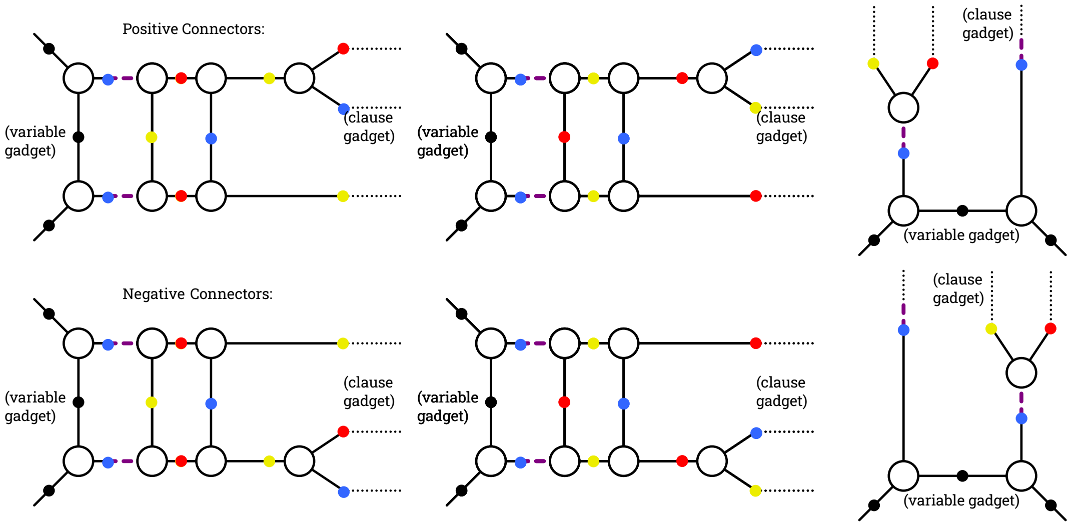


Figure 6 All positive and negative connectors of the reduction in Figure 7.

A Directed 3-Cycle Cover

We reduce from directed 3-cycle cover, disallowing pairs of vertices with both edges between them, which we show is NP-hard in the following result. This result is inspired by problem [20, GT11: Partition Into Triangles], but generalized to directed graphs. We require an additional constraint as well, that there do not exist any two cycles in our graph.

A 3-cycle cover has exactly $\frac{n}{3}$ cycles, so we must design our reduction to have exactly that number of complexes in the minimum free energy structure. To address this, we must make sure that complexes representing 3-cycles are the smallest cycles in our system. This is true if our graph does not contain any 2-cycles, which requires that the graph not contain any doubly covered edges. We now prove NP-hardness and some technical lemmas for our reduction, with variable n denoting the number of vertices in the graph and m denoting the number of edges.

▶ **Theorem 24.** *Directed 3-Cycle Cover is NP-hard even on graphs without any 2-cycles.*

Proof. Planar 3DM [17] is NP-hard even when there are no faces of size 3 without a set. It turns out that mimicking the reduction by Dyer and Frieze is enough. Indeed, this reduction

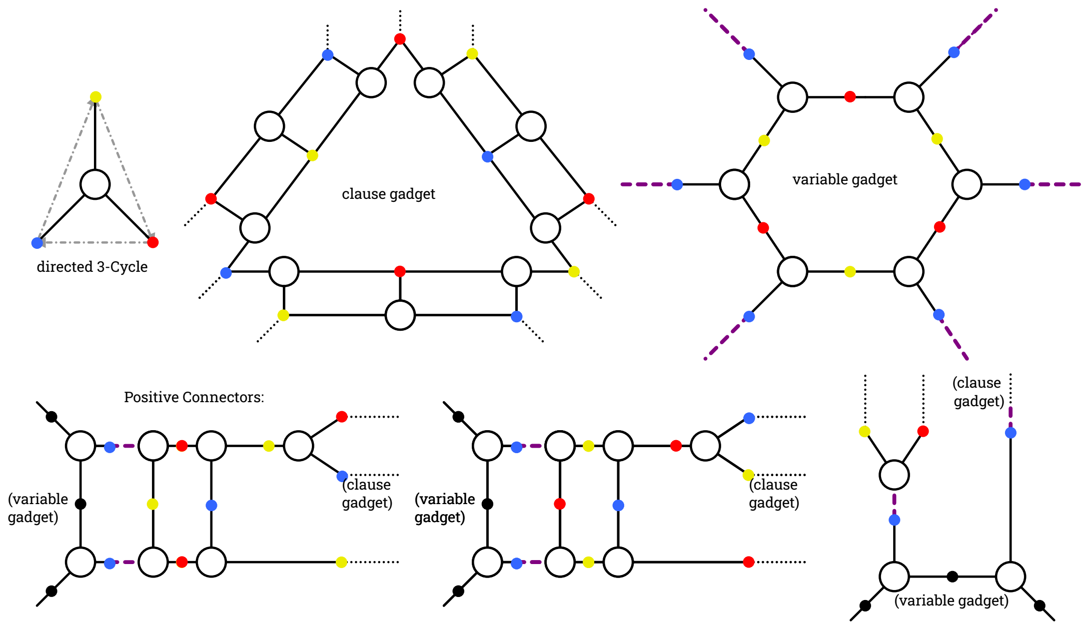


Figure 7 Reduction from 1-in-3SAT to Directed 3-Cycle Cover, which borrows from the reduction to 3DM [17]. (Top): Notation of directed 3-cycles, as well as clause gadgets and variable gadgets. (Bottom): Connectors from positive variable appearances to the clause gadgets. Negative connectors are obtained by swapping the branches going into the clause gadget (see Figure 6 for details). Correctness follows from [17] and the observation of two properties. See the proof of Theorem 24.

does not require planarity but rather preserves it, i.e., the gadgets still work in the non-planar setting. We summarize their gadgets in Figure 7. Their gadgets are taken as is and use colors to specify directed edges, which are fixed from blue to yellow, yellow to red, and red to blue. The main observations we obtain from these gadgets are that (1) there is no 2-cycle in the reduction and (2) there is no new 3-cycle that is not given but can be constructed from a combination of given 3-cycles. Indeed, (1) can't occur as we only construct directed edges from vertices of color blue to yellow, yellow to red, and red to blue. So, the corresponding inverse edge can never exist. The only possibility for (2) is depicted in Figure 5 (left), which would be the case if we combined three different 3-cycles. However, to address this, we require each face uses three differently colored nodes (see Figure 5 (right)), which is not possible in [17] (see also Figure 7). ◀

▶ **Lemma 25.** *Every 3-cycle cover algorithm on directed graphs with n vertices, even on graphs which do not contain any 2-cycles, has a runtime $2^{\Omega(n)}$ unless ETH fails.*

Proof. We first note that the reduction from 3SAT to 1-in-3SAT [42] only increases the number of variables by a linear amount. We then track the chain of reductions from 1-in-3SAT, to 3DM [17], to 3-Cycle cover (Thm. 24) and show the number of vertices in the cycle cover graph is linear in the number of 1-in-3SAT variables, as we do require the reduction to preserve planarity. This preserves the $2^{\Omega(n)}$ lower bound under ETH [26] from SAT to 3-cycle cover. We also note that a $2^{\Omega(n)}$ ETH lower bound for 3DM was shown in [3]. ◀

▶ **Lemma 26.** *All secondary structures achieve at most $2n$ bonds.*

Proof. Each bond in any secondary structure must include a domain from one of the vertex species, and each species has 2 such domains, so the total number of bonds is at most $2n$. ◀

▶ **Lemma 27.** *A 3-cycle secondary structure (if it exists) has* $2n$ *bonds and* $m - \frac{2n}{3}$ *distinct complexes.*

Proof. Each domain of each vertex species is bonded to an edge species in a 3-cycle secondary structure, which implies the structure achieves $2n$ bonds. For the number of distinct complexes, we can count them by including the number of cycles ($\frac{n}{3}$) plus the number of remaining edge species. Since each cycle complex absorbs exactly 3 edge species, the number of remaining edge species is $m - n$, yielding a total of $m - \frac{2n}{3}$ total distinct complexes. ◀

▶ **Lemma 28.** *Any secondary structure with* $2n$ *bonds that is not a 3-cycle secondary structure has less than* $m - \frac{2n}{3}$ *distinct complexes.*

Proof. Consider a secondary structure of $2n$ bonds that is not a 3-cycle secondary structure. Note that each vertex species must be bonded to exactly 2 edge species to achieve $2n$ bonds. Let $d + r$ denote the number of connected complexes in the structure that contain at least one vertex species, with r specifically denoting the number of such complexes that form a connected cycle, and d denoting the number of those that do not.

For each of the r complexes that form a closed cycle of bonds, the number of edge species included in the complex is the same as the number of vertices in the complex, whereas, for each of the d non-cycle complexes, the edge count is one more than the number of vertices in the cycle. Therefore, the total number of edge species that are bonded to one of these complexes is $n + d$. The total number of complexes in the secondary structure can be calculated by including the number of complexes that absorb the vertex species $(d + r)$ plus the number of remaining (unbonded) edge species $(m - n - d)$, for a total of $(d + r) + (m - n - d) = m - n + r$. If this secondary structure is not a 3-cycle structure, then $r < \frac{n}{3}$, and so this total is less than $m - \frac{2n}{3}$. ◀

▶ **Lemma 29.** *If the associative free energy* $0 < \Delta G^{\text{assoc}} < 1$*, then the minimum free energy secondary structure has* $2n$ *bonds.*

Proof. Any secondary structure with fewer than $2n$ bonds would have two separate complexes with complementary, unbonded domains. A new configuration could, therefore, be constructed by combining these two complexes through this pair of domains and increasing both the bond count and complex count by 1. Since $\Delta G^{\text{assoc}} < 1$, this new secondary structure would have less free energy than the original structure, implying only a maximal $2n$ bond secondary structure could be the minimum energy structure. ◀

B Additional Figures

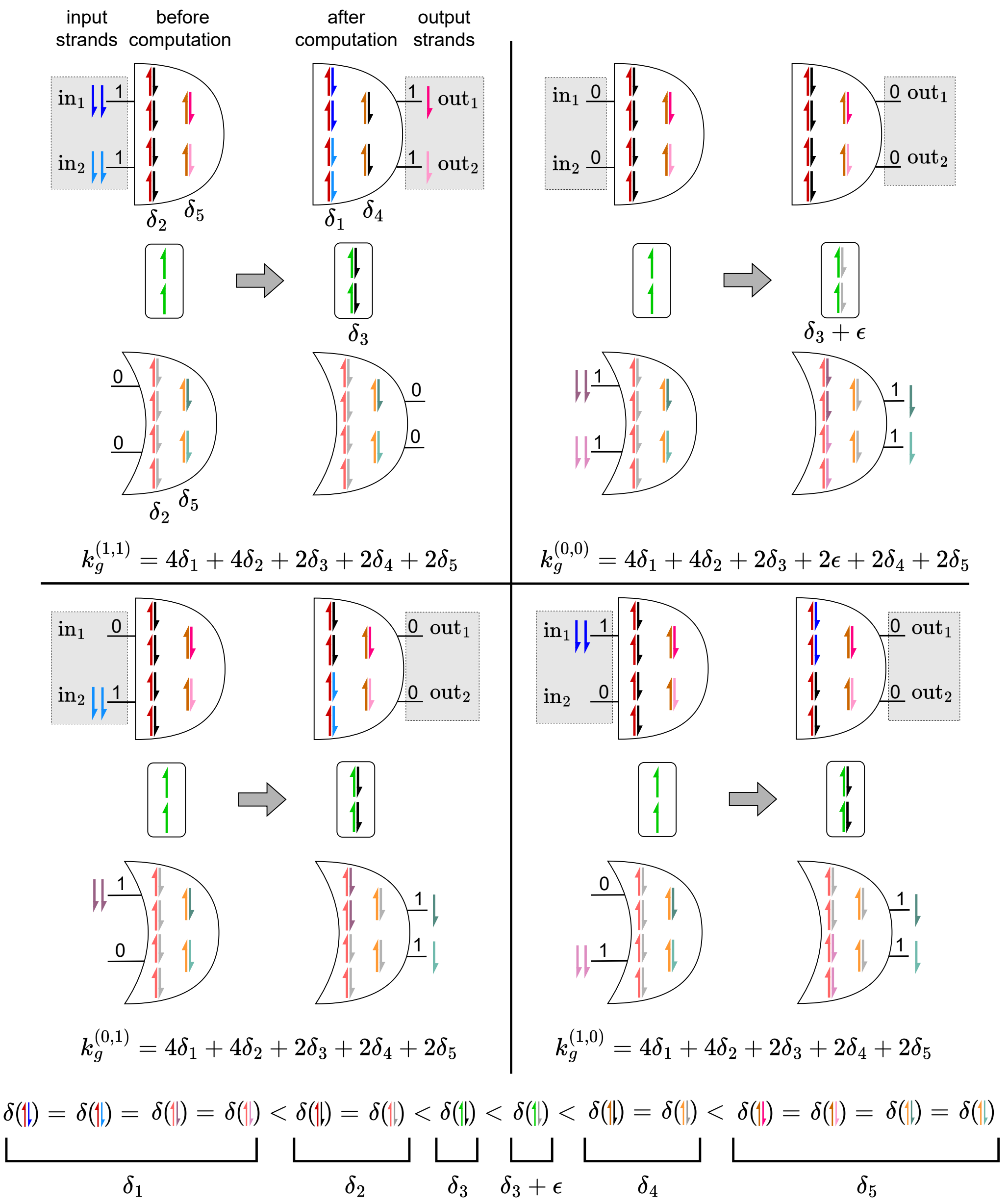


Figure 8 Gate gadget for proof of Theorem 18, showing the design for simulating a circuit consisting of a single AND gate (i.e. depth 1). Simulation of a single AND gate with input bits $\text{in}_1, \text{in}_2 \in \{0, 1\}$ and output bits $\text{out}_1, \text{out}_2 \in \{0, 1\}$. Panels in row-major order respectively show input bit pair 11, 00, 01, and 10. Intuitively, the gadget simulates AND in a dual-rail fashion using three components: an AND component, plus two components that work together to act as a dual to the AND: a small intermediate component (green strands) and an OR component. Together, with a dual rail encoding of the input bits, the three components work to keep the gate energy ($\Delta G()$ almost constant (i.e. constant up to $-\epsilon$). To simulate an OR gate, the same gadget is used, except that inputs are flipped, and the output comes from the OR instead of the AND component. $k_g^{(x,y)}$ denotes the MFE of gate g with input (x, y).

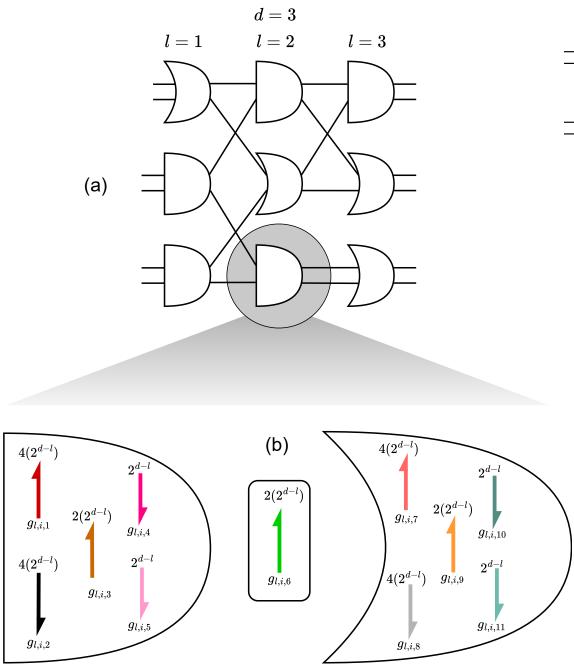


Figure 9 Gate gadget for proof of Theorem 18. (a) An example monotone, fanin-2, fanout-2 circuit C, with a 3×3 layout of AND and OR gates. (b) Design for an arbitrary strand gadget simulating the highlighted AND gate $g_{\ell,i}$ (the ith gate at layer ℓ in C). This gate gadget has 11 strand types named $g_{\ell,i,1}$ to $g_{\ell,i,11}$ that are unique to that gate gadget (they appear in no other gate gadget), with the counts of each strand type shown directly above, as a superscript to the strand type.

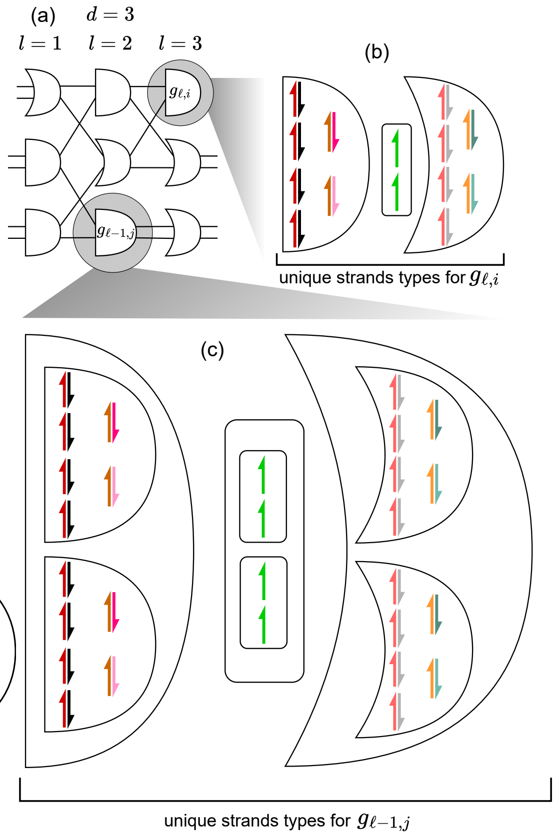


Figure 10 Recursive nature of the construction in the proof of Theorem 18. (a) An example monotone, fanin-2, fanout-2 circuit C, with a 3×3 layout of AND and OR gates. (b) Design for the strand gadget simulating the highlighted AND gate $g_{\ell,i}$ at the output layer. (c) Design the strand gadget simulating the highlighted AND gate $g_{\ell-1,j}$ in the layer just before the output layer. Note that the 11 strand types in each outlined gate gadget are unique to that outlined gate gadget, despite colour-repetition between gadgets $g_{\ell,i}$ and $g_{\ell-1,j}$.