Programming biomolecules that fold greedily during transcription

Cody Geary[1], Pierre-Étienne Meunier[2], Nicolas Schabanel[3], and Shinnosuke Seki[4]

1 **California Institute of Technology, Pasadena, CA, USA.** codyge@gmail.com.
2 **Department of Computer Science, Aalto University, Finland and Aix Marseille Université, CNRS, LIF UMR 7279, 13288, Marseille, France.** http://users.ics.aalto.fi/meunier/
3 **CNRS, Université Paris Diderot, France and IXXI, Université de Lyon, France.** http://www.irif.univ-paris-diderot.fr/users/nschaban/
4 **University of Electro-Communications, Tokyo, Japan.** http://kjk.office.uec.ac.jp/Profiles/69/0006845/prof_e.html

Abstract

We introduce and study the computational power of Oritatami, a theoretical model to explore greedy molecular folding, by which the molecule begins to fold before waiting the end of its production. This model is inspired by our recent experimental work demonstrating the construction of shapes at the nanoscale by folding an RNA molecule during its transcription from an engineered sequence of synthetic DNA. While predicting the most likely conformation is known to be NP-complete in other models, Oritatami sequences fold optimally in linear time.

An important challenge of this model, also encountered in experiments, is to get a single sequence to fold into different shapes, depending on the surrounding molecules. Another big challenge is that not all parts of the sequence are meaningful for all possible inputs; we must hence use such "non-coding parts", in order for them not to interfere with the rest of the sequence.

Next, we introduce general design techniques to solve these challenges and program molecules. Our main result in this direction is an algorithm in time linear in the sequence length, that finds a rule for folding the sequence deterministically into a prescribed set of shapes depending of its environment. This shows the corresponding problem is fixed-parameter tractable, although we also prove it NP-complete in the number of possible environments.

Keywords and phrases Natural computing, Self-Assembly, Molecular Folding

1 Introduction

The process by which one-dimensional sequences of nucleotides or amino-acids acquire their complex three-dimensional geometries, which are key to their *function*, is a major puzzle of biology today. In particular, the problem of predicting how proteins fold is a major source of interest. Indeed, understanding molecular folding will not only shed light on the origin and functions of the molecules existing in nature, it will also enable us to *control* the process more finely, and engineer artificial molecules with a wide range of uses, from performing missing functions inside living organisms, to producing precisely targeted drugs.

Previous nano-engineering with biomolecules included DNA self-assembly, which gave rise to an impressive number of successful experimental realizations, from arbitrary 2D shapes [22] to molecule cyclic machines [27], counters [11]. First pioneered by Seeman [23], DNA nanotechnologies only really started to take off once a computer science model was devised by Winfree [25] to *program* molecular self assembly in a computer science way.

Since then, many models have been designed to refine different features of experiments: hierarchical self-assembly [5, 6], modeling the absence of a *seed*, kinetic tile assembly [25, 13],

3D and probabilistic tile assembly [7], among others. These models, and the theoretical result we can prove on them, are a fundamental component of the broader field, as they allow us to understand reasons for phenomena, as opposed to experiments, which only allow to tell whether something work or did not.

However, the potential applications of DNA are limited by the high stability of DNA itself, meaning that assemblies don't easily react with other molecules. Moreover, stability also means that this assembly process happens primarily by *annealing*: namely, by heating the molecules up to high temperatures in a precisely controlled environment, and then cooling them down at a precisely controlled rate, which is incompatible with many forms of life.

Other biomolecules, such as RNA and proteins, do not have that constraint. However, their assembly process is harder to program: intuitively, the shape depends on the sequence and the environment, and the sequence is read *linearly*, independent from the environment. This contrasts with more classical programming models such as Turing machines, or even tile assembly, which are able to *jump* to different parts of the program depending on the input.

Another important difficulty is that even predicting the final shape of a sequence is still the center of active research, especially for proteins [28, 17, 18, 21, 16]

In particular, a large body of computer science literature focused on energy optimization, one of the main drivers of folding. For example, in different variants of the *hydrophobic-hydrophilic (HP) model* [9], it has been shown that the problem of predicting the most likely geometry (or *conformation*) of a sequence is NP-complete [24, 20, 3, 4, 8, 2], both in two and three dimensions, and in different variants of the model.

A few years ago, the *kinetics* of folding, which is the step-by-step dynamics of the reaction, has been demonstrated by biochemists to play a fundamental role in the final shape of molecules [15], even more so in the case of RNA [12]. In recent experimental results [14], researchers have even been able to *control* this mechanism to engineer their own shapes out of RNA.

This paper introduces a new model of folding, intended to capture the kinetics of folding and model the experiments in [14]. In particular, it focuses on the *co-transcriptional* nature of folding, which is the fact that, in real conditions, molecules fold while being transcribed (see Figure 1): in computer science terms, the folding process is a *local energy optimization*, or otherwise put, a *greedy algorithm*.

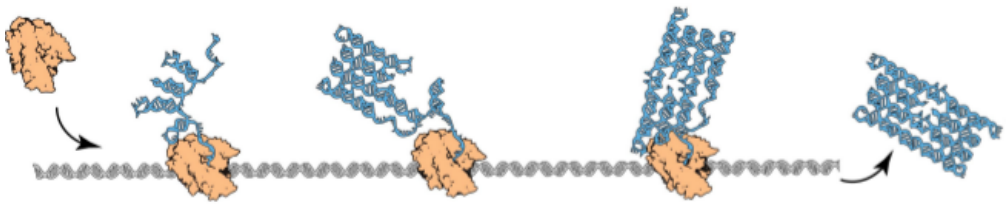


Figure 1 An RNA molecule folding over itself while being transcribed, as the experiments in [14].

The first experimental results have used a standard benchmark: making simple shapes, such as squares (as shown for instance on Figure 2). With the new model introduced in this paper, our goal is twofold: first, explore the engineering possibilities of this mechanism, in order to make arbitrary shapes and structures. Then, the other aim of our study is to understand the complexity of sequence operations, to understand the computational processes which led to the creation of complex molecular networks.

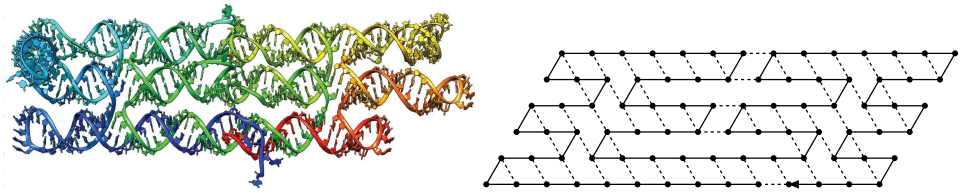


Figure 2 The design of a square, co-transcriptionally folded with RNA, and the corresponding path on the triangular lattice.

Main contributions. In our model, called Oritatami, we consider a sequence of "beads", which are abstract basic components, standing for nucleotides or even sequences of nucleotides (also called *domains*). In Oritatami, only the latest produced beads of the molecules are allowed to move in order to adopt a more favorable configuration. The folding is driven by the respective attraction between the beads.

Our first construction is a *binary counter*. Counters are an essential component of many sophisticated constructions in biological computing, in particular in tile assembly [10, 19]. Counters are also an important benchmark in experiments [11].

▶ **Theorem 1** (Informal statement). *There is a fixed periodic sequence of period 60 which, when started next to another molecule encoding some integer x, written on $2k+1$ bits for some k, folds into a structure encoding $x+1$, $x+2$, ..., $2^{2k+1}-1$, on successive rows of the grid.*

We *prove the correctness* of this construction by designing an abstract module system to handle the complexity of the base mechanism of the model, which is about as low-level as assembly code in more standard computing models.

We then show a generic construction method in this model, which we applied to automate parts of the design of the counter. Moreover, this result helps understanding the computational complexity of sequence programming. Precisely, we prove two results in this direction:

▶ **Theorem 2** (Informal statement). *Designing a single sequence that folds into different prescribed shapes according to the different surrounding molecules, is NP-complete in the number of shapes.*

More surprisingly, it turns out that there is an algorithm to solve this problem in time *linear in the length of the sequence*. This algorithm is also practical, as we were able to use it to find sequences for our main construction:

▶ **Theorem 3** (Informal statement). *The sequence design problem is FPT, and there is an algorithm linear in the length of the sequence.*

2 Model and Main Results

2.1 Model

Oritatami system. Oritatami is about the folding of finite sequences of beads, each from a finite set B of *bead types*, using an attraction rule $\heartsuit$, on the triangular lattice graph $\mathbb{T} = (\mathbb{Z}^2, \sim)$ where $(x,y) \sim (u,v)$ if and only if $(u,v) \in \{(x-1,y),(x+1,y),(x,y+1),(x+1,y+1),(x-1,y-1),(x,y-1)\}$.

A *conformation* c of a sequence $w \in B^*$ is a self-avoiding path of length ℓ labelled by w in $\mathbb{T}$, i.e. a path whose vertices $c_1, \ldots, c_\ell$ are pairwise distinct and labelled by the letters of w. A *partial conformation* of a sequence w is a conformation of a prefix of w. For any partial conformation c of some sequence w, an *elongation* of c by k beads is a partial conformation of w of length $|c| + k$. We denote by $\mathcal{C}_w$ the set of all partial conformations of w. We denote by $c^{\triangleright k}$ the set of all elongations by k beads of a partial conformation c of a sequence w and by $c^{\triangleleft k}$ the singleton containing the prefix of length $|c| - k$ of c.

An *Oritatami system* $\mathcal{O} = (p, \heartsuit, \delta)$ is composed of (1) a (possibly infinite) *primary structure* p, which is a sequence of *beads*, of a type chosen from a finite set B, (2) an *attraction rule*, which is a symmetric relation $\heartsuit \subseteq B^2$ and (3) a parameter δ called the *delay time*.

Given an attraction rule $\heartsuit$ and a conformation c of a sequence w, we say that there is a *bond* between two adjacent positions c_i and c_j of c in $\mathbb{T}$ if $w_i \heartsuit w_j$. The *energy* of a conformation c of w, written $\mathrm{E}(c)$, is the negation of the number of bonds within c: formally, $\mathrm{E}(c) = -|\{(i,j) : c_i \sim c_j,\ j > i+1,\ \text{and } w_i \heartsuit w_j\}|$.

Oritatami dynamics. A *dynamics* for a sequence w is a function $\mathcal{D}_w : 2^{\mathcal{C}_w} \to 2^{\mathcal{C}_w}$ such that for all subset S of partial conformations of length ℓ of w, $\mathcal{D}(S)$ is a subset of the elongations by one bead of the partial conformations in S (thus, partial conformations of length $\ell + 1$).

Given an Oritatami system $\mathcal{O} = (p, \heartsuit, \delta)$ and a *seed conformation* σ of a seed sequence s of length ℓ, the set of partial conformations of the primary structure p at time t under dynamics $\mathcal{D}$ is $\mathcal{D}^t_{sp}(\{\sigma\})$,[1] i.e. the set of all elongations by t beads of the seed conformation prolongated by the primary structure according to dynamics $\mathcal{D}$.

We explore greedy folding dynamics where only the most recently transcribed beads can move, all other beads remain in place. These are controlled by integer parameter δ (in this article, $\delta \leqslant 4$). Several dynamics could model the "greedy" nature of the process. We choose the following dynamics, called the *hasty dynamics*:

The hasty dynamics does not question previous choices but chooses the energy minimal positions for the δ last beads among all elongations of the previously adopted partial conformations. It lets the $\delta - 1$ already placed last beads where they are and abandons the extension of a conformation if no extension with the newly transcribed bead allows to reach a lowest energy conformation available for the δ last beads. Formally, $\mathcal{H}$ starts from a set of partial conformations, elongates each of them by one bead, and keeps the elongated conformations that have minimal energy among those who share the same prefix of length $|\sigma| + t - \delta$:

$$\mathcal{H}(S) = \bigcup_{\gamma \in S^{\triangleleft(\delta-1)}} \left(\operatorname*{arg\,min}_{c \in (S^{\triangleright 1}) \cap (\gamma^{\triangleright \delta})} \mathrm{E}(c) \right)$$

An Oritatami system $\mathcal{O} = (p, \heartsuit, \delta)$ is *deterministic* for dynamics $\mathcal{D}$ and seed σ of sequence s if for all $i \geqslant 1$, the position of the i-th bead of p is deterministic at time $i - 1 + \delta$, i.e. if for all $i \geqslant 1$, $|\{c_{|\sigma|+i} : c \in \mathcal{D}^{i-1+\delta}_{sp}(\{\sigma\})\}| = 1$. We say that $\mathcal{O}$ *stops* at time t with seed σ and dynamics $\mathcal{D}$ if $D^t_{sp}(\{\sigma\}) = \varnothing$ and $\mathcal{D}^z_{sp}(\{\sigma\}) \neq \varnothing$ for $z < t$. Typically, the folding process stops because of geometric obstruction (no more elongation are possible because the conformation gets trapped in a closed area).

[1] Given two words $a, b \in B^*$, we denote by ab their concatenation.

3 Folding a binary counter

The goal of this section is to prove the following theorem:

▶ **Theorem 1.** *There is a 60-periodic primary sequence s such that for any integer n, there is an encoding of n into a seed σ of width $w \leqslant 1 + 10\lfloor \log_2(n) \rfloor$ on row 0, such that s folds into a structure which encodes $n + k$ on row $-6k$, for all integer k such that $n + k < 2^w$. This construction works at $\delta = 4$.*

3.1 General idea of the construction

The counter is folded by folding a periodic sequence of length 60, with bead types 0, 1, ..., 59, into shapes representing values of the counter. Computation happens when the last δ transcribed beads are folded.

Our construction proceeds in zig-zags: the *zig pass*, folding three rows at a time from right to left, computes the next value of the counter. The *zag pass*, folding three rows at a time from left to right, copies that new value, and resumes the construction at the right-hand side of the conformation.

3.1.1 Modules and functions

We decompose the construction into *modules*, each implementing different *functions*. Modules are factors of the sequence. Each period of our structure has four modules: the first and third modules (beads 0-11 and 30-41, respectively), in blue on all the figures, are *half-adder modules*; the second one (beads 12-29), in red on all the figures, is the *left U-turn module*, and the fourth one (beads 42-59) is the *right U-turn module*.

Each module implements a number of *functions*, which map the environment (i.e. the surrounding beads) to a conformation. For instance, Figure 3 shows two functions of the first half-adder module.

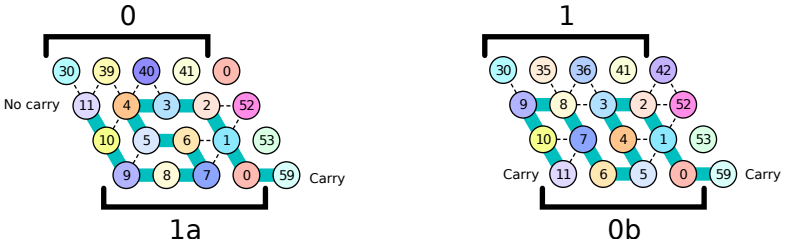


Figure 3 Two runs of the half-adder function of the half-adder module, on different input environments: both start on the bottom row (encoding a carry), and read a different value (0 on the function on the left-hand side, 1 on the function on the right-hand side).

3.1.2 Initial conformation

We start with a seed like the one shown on Figure 4, encoding n in binary (see Figure 4 for an example of the encoding). Reading from left to right, the first three beads are fixed for any number, and then two patterns alternate: one with beads between 30 and 59, the other one with beads between 0 and 29. The alternation starts and ends with beads between 30 and 59.

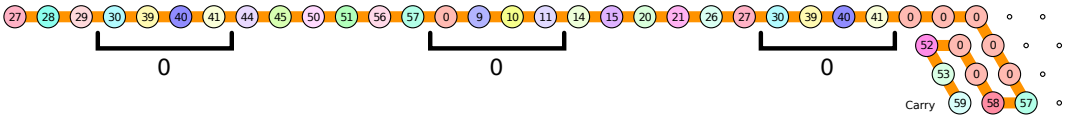


Figure 4 Initial seed, encoding integer 0: from left to right, the first three beads are fixed for any value of the counter, and then two patterns alternate, encoding bit values with either beads 0 to 11, or beads 30 to 41.

3.1.3 The zig pass

The "output" of the zig pass (i.e. the bottom row at the end of the zig pass), is an encoding of the next value of the counter, using a different representation for 0 and 1, called $0a$, $0b$ (for 0), and $1a$, $1b$ for 1. Then, the zag pass rewrites this encoding, translating $0a$ and $0b$ into 0, and $1a$ and $1b$ into 1 (see Figure 8).

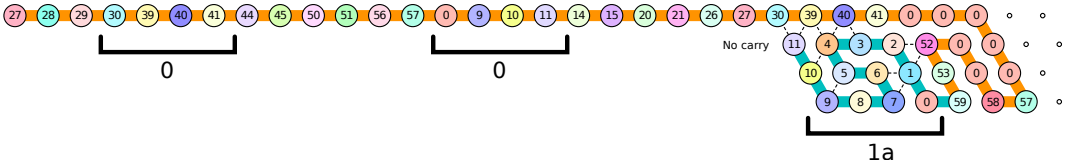


Figure 5 The folding of the first module: starting with a carry, encoded by the position of the first bead (on the bottom row), this module "reads" a 0 from the seed by binding to the seed, and outputs "$1a$", an encoding of 1 after the zig pass, and no carry, encoded by the position of the last bead (on the top row of the module).

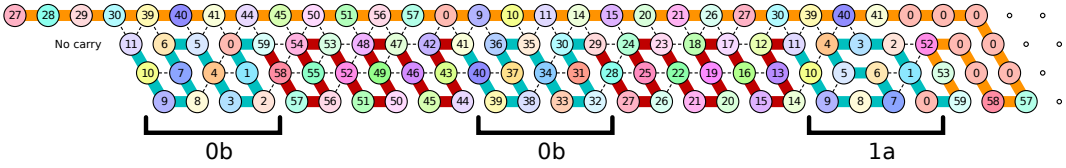


Figure 6 After repeating an odd number of times, the leftmost half-adder module is the first module in the sequence. Other half-adders may compute either $0a$ or $0b$ (the two encodings of 0 after the zig pass), or $1a$ or $1b$ (the encodings of 1 after the zig pass).

3.1.4 The zag pass

The zag pass progresses from left to right, rewriting $0a$ and $0b$ into 0, and $1a$, $1b$ into 1. It starts and ends with the second half-adder module (beads 30-41).

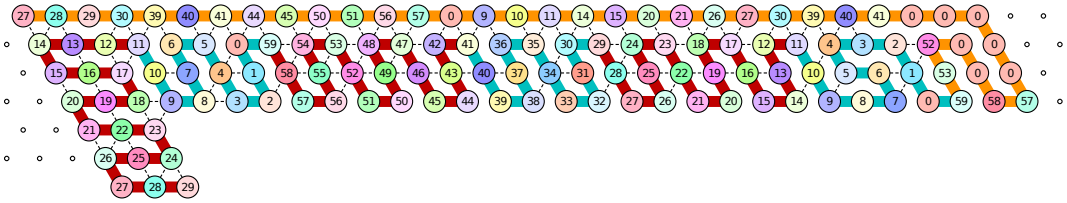


Figure 7 In our construction, the leftmost three beads of any conformation are different from the other beads the left U-turn module binds to inside the zig or zag pass: when the left U-turn module folds next to these bead types, it "triggers" the production of an actual U-turn.

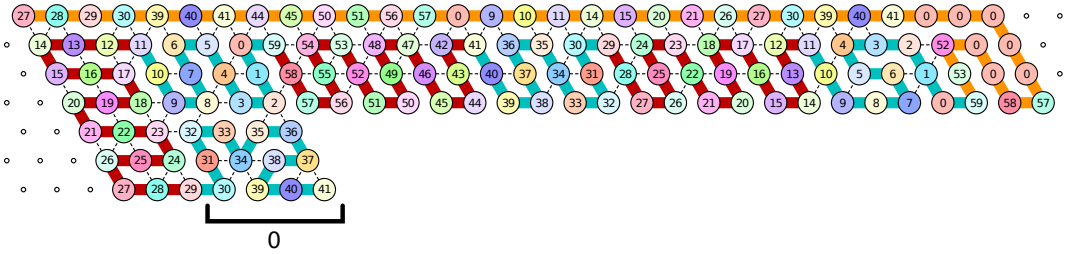


Figure 8 During the zag pass, all modules start from the bottom row, translating $0a$ and $0b$ to 0, and $1a$, $1b$ to 1.

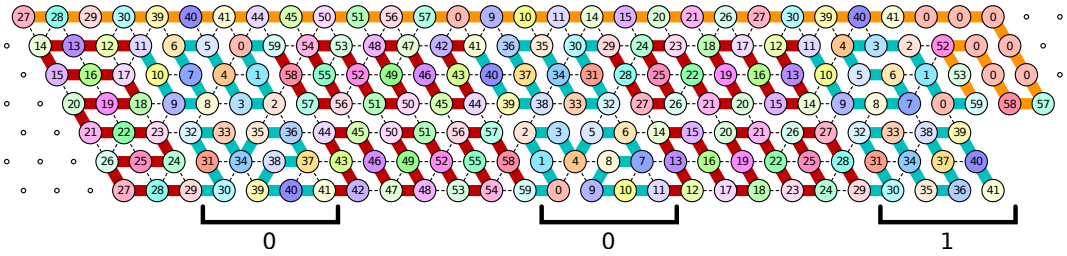


Figure 9 At the end of the first zag pass, all zag encodings of 0 (i.e. $0a$ and $0b$) are rewritten as zig encodings of 0, and all zag encodings of 1 (i.e. $1a$ and $1b$) are rewritten as zig encodings of 1.

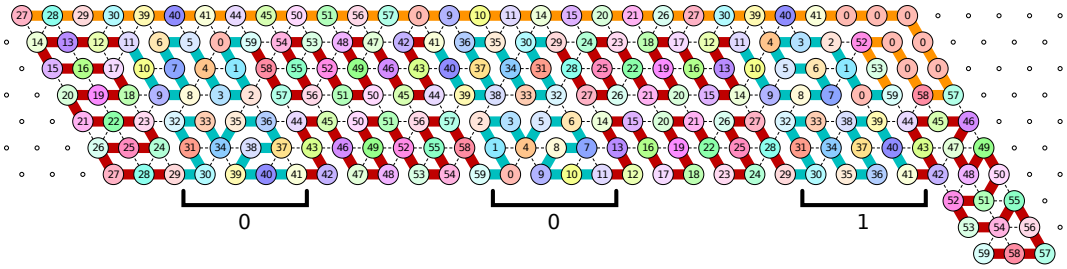


Figure 10 Finally, after the first zag pass, the right U-turn module folds into an actual U-turn, with the same environment as in the seed: the next zig pass can start.

3.2 Proof

We first prove (in Lemma 4) that if the specifications of all modules are respected, the final conformation we get contains the encodings of all successive values of the counter. This lemma assumes that values of the counter before zig passes are encoded in a form called the *zig encoding* of the value, and values before zag passes are encoded as *zag encodings*. Zig- and zag-encodings are formally defined by Definition 5.

▶ **Lemma 4.** *If all the modules implement all their functions, and the seed is a zig encoding of* 0 *of length some* w*, then for all integer* $0 \leqslant k < 2^w$*, the end conformation contains a zig encoding of* k *on row* $-6k$*.*

Proof. We first need to list the modules' conformations:

- The first half-adder module has eight conformations in the zig pass:
  - During the zig pass, the first module starts with a carry, and reads $r \in \{0, 1\}$, outputting $r + 1 \mod 2$. Two conformations.
  - Each subsequent module of the zig pass reads a zig encoding $r \in \{0, 1\}$, with or without a carry, and outputs $r + 1 \mod 2$. Note that the environment on the right of the module is different from the first module (see Figure 11). This also yields four conformations.
  - The last module always has carry 0 except on the last value. Therefore, it always reads 0, and thus has two conformations.

  And four in the zag pass, as it needs to read all four different zag bit encodings. In the zag pass, this module reads zag encodings of a bit, and output the corresponding zig encoding.
- The second half-adder module has four conformations in the zig pass (as it never starts nor ends the row), and also six conformations in the zag pass: four as the leftmost module (one for each zag encoding), and four in other parts (one for each zag encoding).
- The two other modules have:
  - Four conformations each in the zig pass, depending on the carry and the bead in the top right corner (which can be of two different types for each module).
    Each of these conformations ends on the row where it started.
  - One U-turn conformation each.

If these functions can be implemented, then after a zig pass starting on the zag encoding of some integer n, the bottom row contains a zag encoding of $n + 1$ (this can be proven by induction on the number of bits). Moreover, after a zag pass, the bottom row contains a zig encoding of $n + 1$. Finally, if the U-turns can be implemented, the zig and zag rows alternate until the counter overflows (i.e. reaches 2^w). ◀

▶ **Definition 5.** A *non coding zig segment* is a word of length 6 over the alphabet of beads. There are two non-coding words:

- 44, 45, 50, 51, 56, 57.
- 14, 15, 20, 21, 26, 27.

A *zig bit encoding* is a word of length 4 over the alphabet of beads. There are four zig bit encodings:

- 0, 9, 10, 11, encoding 0.
- 30, 39, 40, 41, encoding 0.
- 0, 5, 6, 11, encoding 1.
- 30, 36, 36, 41, encoding 1.

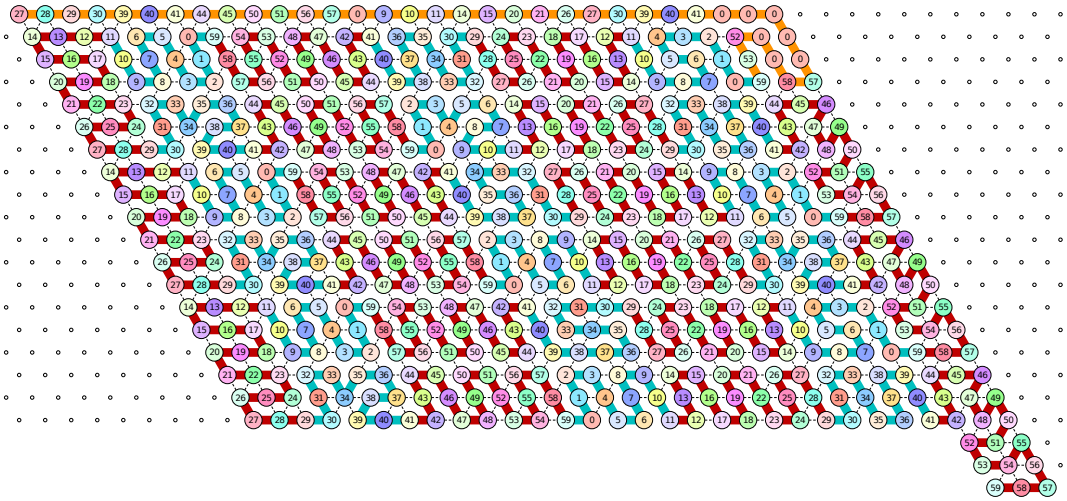


Figure 11 Three successive iterations of the counter, starting from the zig encoding of 0 on 3 bits (top row, in orange), and ending with the zig encoding of 3 on 3 bits (bottom row).

A *zag bit encoding* is a word of length 4 over the alphabet of beads. There are eight zag bit encodings:

- 9, 8, 3, 2, encoding 0.
- 11, 6, 5, 0, encoding 0.
- 39, 38, 33, 32, encoding 0.
- 41, 36, 35, 30, encoding 0.
- 9, 8, 7, 0, encoding 1.
- 9, 8, 7, 6, encoding 1.
- 39, 38, 37, 30, encoding 1.
- 39, 38, 37, 36, encoding 0.

A *zig encoding* of integer n on $w \geqslant \log_2(n)$ bits, where w is an odd integer, is a row of beads of length $10w$, of the following types, read from left to right:

- 27, 28, 29
- $w-1$ times the following word: one non-coding segment, and one bit encoding. The $n-1$ encodings alternate between beads from 30 to 59, and beads 0 to 29, starting with 30-59.
- one bit encoding with beads $30-41$.

▶ **Lemma 6.** *Each of the modules folds correctly in each of the input environments required for the counter: the half-adders correctly compute a half-adder, and the U-turns fold into rectangles inside passes, or into U-turns at the ends of passes.*

Proof. The full rule ♡ is given on Figure 12. We need to show that the hasty dynamics produces the expected conformations on the zig and zag encodings of every bit of the counter.

It is enough to show that all the cases described in Lemma 4 are handled. For instance, running the 5-bits counter from 0 until it overflows presents all the 12 possible input environments for the first half-adder module 0-11 as highlighted in yellow and labelled from A to L on Figure 13.

We prove that the folding of each module in each environment is correct by computing the tree of all possible foldings of each elongation of the current conformations by one bead, retaining for the next level of the tree, only the conformations (in bold) with the maximum

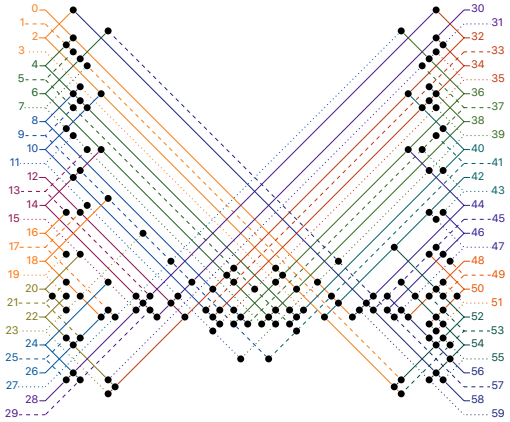


Figure 12 A representation of the full rule for the counter. Bead types are written on the sides of the diagram. From each bead type, zero, one or two lines can start. For any pair (b_0, b_1) of bead types, $b_0 \heartsuit b_1$ if and only if the line starting from b_0 intersects the line starting from b_1 with a black circle. Note that this representation is symmetric.

number of new bonds. We obtain a tree whose leaves are the resulting conformations of the folding. We call these trees *folding certificates*. Due to space constraint, we only provide all the folding certificate trees for the first half-adder module 0-11. These trees can be found in Appendix A.

All the missing folding certificates for the three other modules can be downloaded from the website [1]. ◀

4 Rule design is NP-hard and FPT

An important problem related to our main constructions (Section 3) is the problem of finding an attraction rule such that a primary structure folds into its correct functions. This section introduces an algorithmic approach to this problem, called the *rule design problem*, and specified as follows:

INPUT:	a delay time δ, a list of $n > 0$ seeds $\sigma_1, \sigma_2, \ldots, \sigma_n$, and a list of n conformations $c_1, c_2, \ldots, c_n$ of the same length l
OUTPUT:	an attraction rule $\heartsuit$ such that for all $i \in \{1, 2, \ldots, n\}$, Oritatami system $\mathcal{O}_i = (s, \sigma_i, \heartsuit, \delta)$ deterministically folds into conformation c_i, where s is the sequence of length l such that for all $i \in \{1, 2, \ldots, l\}$, $s_i = i$.

4.1 NP-completeness

We begin by showing that the rule design problem is NP-complete:

▶ **Theorem 2.** *For any positive delay time, the rule design problem is NP-complete.*

Proof. We reduce from 3-SAT with n variables and m clauses to the rule design with $n + m$ different conformations to be uniquely folded simultaneously. Let $x_0, x_1, \ldots, x_{n-1}$ be the variables, and $F_0, F_1, \ldots, F_{m-1}$ be the clauses of a 3-SAT formula.

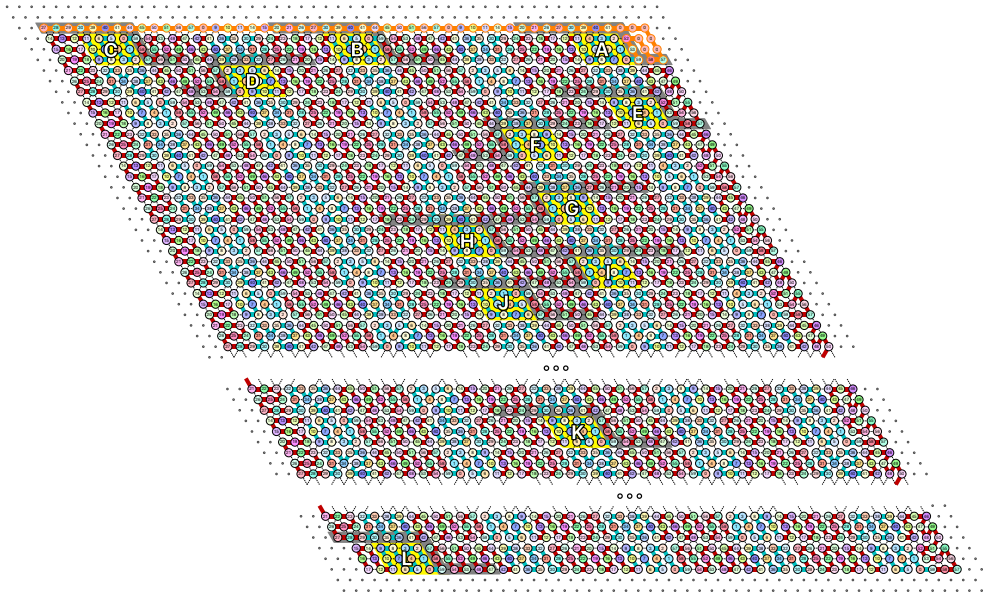


Figure 13 A full run of the 5-bits counter, starting from 0. The conformations highlighted in yellow are all possible conformations of the first half-adder modules. The trees showing the uniqueness of these conformations in all possible environments are given in Appendix A. On this figure, environments at the beginning of the folding of the module are highlighted in gray.

We will encode all $2n$ possible literals by distinct bead types in seeds, one for each possible literal. Figure 14 shows the encoding of a clause of the form $l_i \wedge l_j \wedge l_k$, where l_i, l_j and l_k are literals.

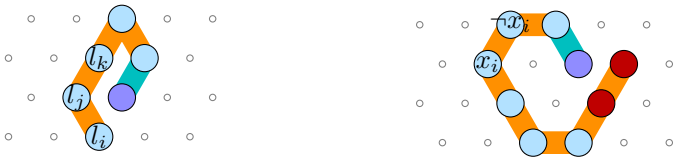


Figure 14 The left-hand side drawing shows the encoding of a clause by a target seed (in blue and orange) and conformation (in purple): if there is at least one attraction between the orange bead and the seed, exactly this conformation is produced. Else, other conformations (not in the targets) are producible. The right-hand side drawing shows the set of target seed (in blue and orange) and conformation (the purple bead) makes sure that x_i and $\neg x_i$ are not both picked by the rule at the same time.

Then, if a rule folds all conformations obtained by our reduction correctly, we will set $x_i = \texttt{true}$ whenever literal x_i is attracted to any bead type in the rule, and $x_i = \texttt{false}$ else.

However, we need to enforce that for all i, x_i and $\neg x_i$ are not both attracted to other beads[2]. We add another n targets to make sure that x_i and $\neg x_i$ are not both chosen by the rule: in the target conformation shown on Figure 14, the first bead produced has two neighboring beads from the seed. If the first bead were attracted to both x_i and $\neg x_i$, another

[2] If neither is selected, setting x_i to either true or false does not change the result.

conformation, not in the targets, would be producible, with the first bead next to x_i and $\neg x_i$.

Finally, this proof works for delay time 1. Extending it a larger delay time $\delta > 1$ can be done by adding a bead in the seed, δ points away from the first bead produced, for each of the seeds. Then, add a straight line of length $\delta - 1$ to that point in the target conformation. ◀

4.2 An FPT algorithm

▶ **Theorem 3.** *The rule design problem with n target conformations, each of length l, and delay time δ is fixed-parameter tractable, as it can be solved in time and space complexity $l2^{9n^2(\delta+1)^4}$.*

Proof. The FPT algorithm solves reachability in a graph of partial rules: for each $i \in \{0, 1, \ldots, l-\delta\}$, let B_i be the set of all bead types that beads $i, i+1, \ldots, i+\delta$ can be adjacent to in all target conformations, assuming all beads from 0 to i (inclusive) are placed correctly in each of the target conformations.

Then, let $\mathcal{R}_i$ be the set of all possible symmetric relations on $\{i, i+1, \ldots, i+\delta\} \cup B_i$. We say that two binary relations $R \in \mathcal{R}_i$ and $T \in \mathcal{R}_{i+1}$ are *compatible* if $R \cap I = T \cap I$, where $I = (\{i, i+1, \ldots, \delta+1\} \cup B_i \cup B_{i+1})^2$.

In other words, R and T are compatible if and only if they are equal on their restriction to their common beads in the primary structure.

We then solve reachability from all rules of $\mathcal{R}_0$ to any rule of $\mathcal{R}_{l-\delta}$, in the graph whose set of vertices is $V = \bigcup_{i=0}^{l-\delta} \mathcal{R}_i$, and E is the compatibility relation on V.

Now, in each target conformation, there are at most $3(\delta+1)^2$ beads in a radius δ, including beads of the primary structure. Therefore, for all i, $|\{i, i+1, \ldots, i+\delta\} \cup B_i| \leqslant 3n(\delta+1)^2$, and hence, $|\mathcal{R}_i| \leqslant 2^{9n^2(\delta+1)^4}$. This means that we are solving reachability in a graph of size at most $l2^{9n^2(\delta+1)^4}$, hence our result[3] ◀

5 Perspectives

The purpose of our new model is not to be entirely accurate with respect to phenomena observed in nature, but instead to start developing an intuition about the *kind of problems* that need to be solved in order to engineer RNA shapes, and later, even proteins.

In the future, a number of extensions of this model seem natural. In particular, extending it with a more realistic notion of *thermodynamics and molecular agitation*. Using existing works in molecular dynamics [26], would allow to explore a stochastic optimization process instead of a deterministic one.

Acknowledgments

The authors thank Abdulmelik Mohammed, Andrew Winslow, Damien Woods for discussions and encouragements. Cody Geary is a Carlsberg Postdoctoral Fellow supported by the Carlsberg Foundation. Nicolas Schabanel is supported by Grants ANR-12-BS02-005 RDAM and IXXI-MOLECAL. The work of Shinnosuke Seki is in part supported by the Academy of Finland, Postdoctoral Researcher Grant 13266670/T30606.

[3] This space complexity might not make the algorithm usable on actual machines, but by generating the $\mathcal{R}_i$s lazily, we were able to compute rules for the counter.

References

1 The folding certificates for all modules. `http://users.ics.aalto.fi/meunier/oritatami_all_modules.pdf`.

2 O. Aichholzer, D. Bremner, E. D. Demaine, H. Meijer, V. Sacristán, and M. Soss. Long proteins with unique optimal foldings in the H-P model. *Computational Geometry*, 25(1–2):139–159, 2003.

3 J. Atkins and W. E. Hart. On the intractability of protein folding with a finite alphabet of amino acids. *Algorithmica*, 25(2–3):279–294, 1999.

4 Bonnie Berger and Tom Leighton. Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. *Journal of Computational Biology*, 5(1):27–40, 1998.

5 Sarah Cannon, Erik D. Demaine, Martin L. Demaine, Sarah Eisenstat, Matthew J. Patitz, Robert Schweller, Scott M. Summers, and Andrew Winslow. Two hands are better than one (up to constant factors). In *STACS: Proceedings of the Thirtieth International Symposium on Theoretical Aspects of Computer Science*, pages 172–184, 2013. Arxiv preprint: `1201.1650`.

6 Ho-Lin Chen and David Doty. Parallelism and time in hierarchical self-assembly. In *SODA 2012: Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1163–1182. SIAM, 2012.

7 Matthew Cook, Yunhui Fu, and Robert T. Schweller. Temperature 1 self-assembly: Deterministic assembly in 3D and probabilistic assembly in 2D. In *SODA 2011: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2011. Arxiv preprint: `arXiv:0912.0027`.

8 Pierluigi Crescenzi, Deborah Goldman, Christos Papadimitriou, Antonio Piccolboni, and Mihalis Yannakakis. On the complexity of protein folding. *Journal of computational biology*, 5(3):423–465, 1998.

9 K.A. Dill. Theory for the folding and stability of globular proteins. *Biochemistry*, 24(6):1501–1509, 1985.

10 David Doty, Jack H. Lutz, Matthew J. Patitz, Robert T. Schweller, Scott M. Summers, and Damien Woods. The tile assembly model is intrinsically universal. In *FOCS 2012*, pages 439–446, October 2012.

11 Constantine Glen Evans. *Crystals that count! Physical principles and experimental investigations of DNA tile self-assembly*. PhD thesis, California Institute of Technology, 2014.

12 Kirsten L. Frieda and Steven M. Block. Direct observation of cotranscriptional folding in an adenine riboswitch. *Science*, 338(6105):397–400, 2012.

13 Kenichi Fujibayashi, David Yu Zhang, Erik Winfree, and Satoshi Murata. Error suppression mechanisms for dna tile self-assembly and their simulation. *Natural Computing: an international journal*, 8(3):589–612, 2009.

14 Cody Geary, Paul W. K. Rothemund, and Ebbe S. Andersen. A single-stranded architecture for cotranscriptional folding of RNA nanostructures. *Science*, 345:799–804, 2014.

15 Boyle J, Robillard G, and Kim S. Sequential folding of transfer RNA. a nuclear magnetic resonance study of successively longer tRNA fragments with a common 5' end. *J Mol Biol*, 139:601–625, 1980.

16 Hosna Jabbari and Anne Condon. A fast and robust iterative algorithm for prediction of rna pseudoknotted secondary structures. *BMC bioinformatics*, 15(1):147, 2014.

17 D. H. Mathews, M. D. Disney, J. L. Childs, S. J. Schroeder, M. Zuker, and D. H. Turner. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *PNAS*, 101:7287–7292, May 2004.

18 D.H. Mathews. Revolutions in rna secondary structure prediction. *Journal of molecular biology*, 359(3):526–32, 2006.

19 Pierre-Étienne Meunier, Matthew J. Patitz, Scott M. Summers, Guillaume Theyssier, Andrew Winslow, and Damien Woods. Intrinsic universality in tile self-assembly requires cooperation. *SODA 2014: Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2014.

20 M. Paterson and T. Przytycka. On the complexity of string folding. In F. Meyer and B. Monien, editors, *ICALP 1996*, volume 1099 of *LNCS*, pages 658–669. Springer Berlin Heidelberg, 1996.

21 Elena Rivas. The four ingredients of single-sequence rna secondary structure prediction. a unifying perspective. *RNA Biol*, 10(7):1185–1196, Jul 2013. 23695796[pmid].

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

23 Nadrian C. Seeman. Nucleic-acid junctions and lattices. *Journal of Theoretical Biology*, 99:237–247, 1982.

24 R. Unger and J. Moult. Finding the lowest free energy conformation of a protein is an NP-hard problem: proof and implications. *Bulletin of Mathematical Biology*, 55(6):1183–1198, 1993.

25 Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, Caltech, June 1998.

26 Damien Woods, Ho-Lin Chen, Scott Goodfriend, Nadine Dabby, Erik Winfree, and Peng Yin. Active self-assembly of algorithmic shapes and patterns in polylogarithmic time. In *Proceedings of ITCS 2013: Innovations in Theoretical Computer Science*, pages 353–354, 2013.

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

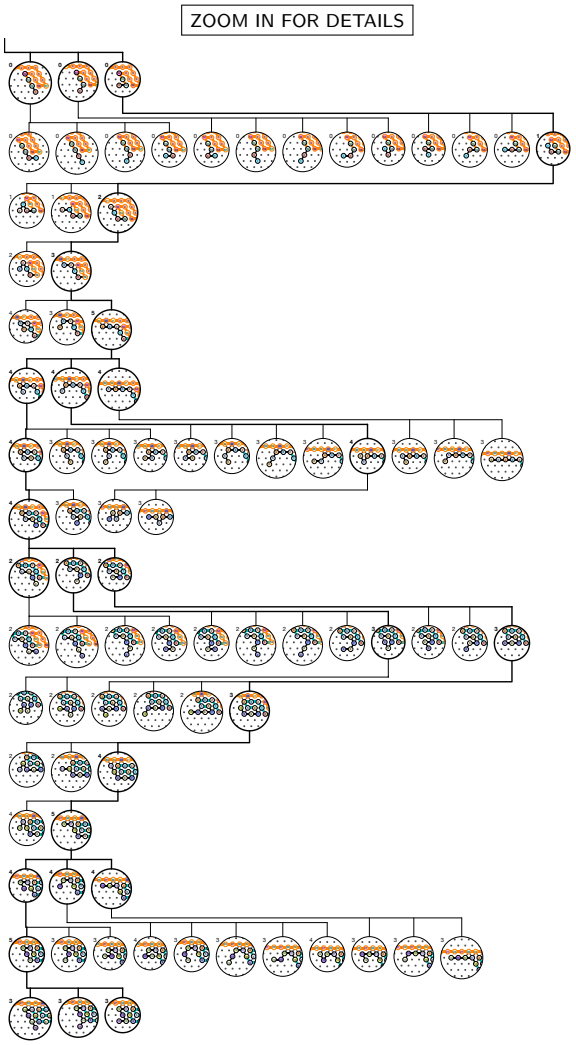
28 Michael Zuker and David Sankoff. Rna secondary structures and their prediction. *Bulletin of Mathematical Biology*, 46(4):591–621, 1984.

A Certificates for the first half-adder module

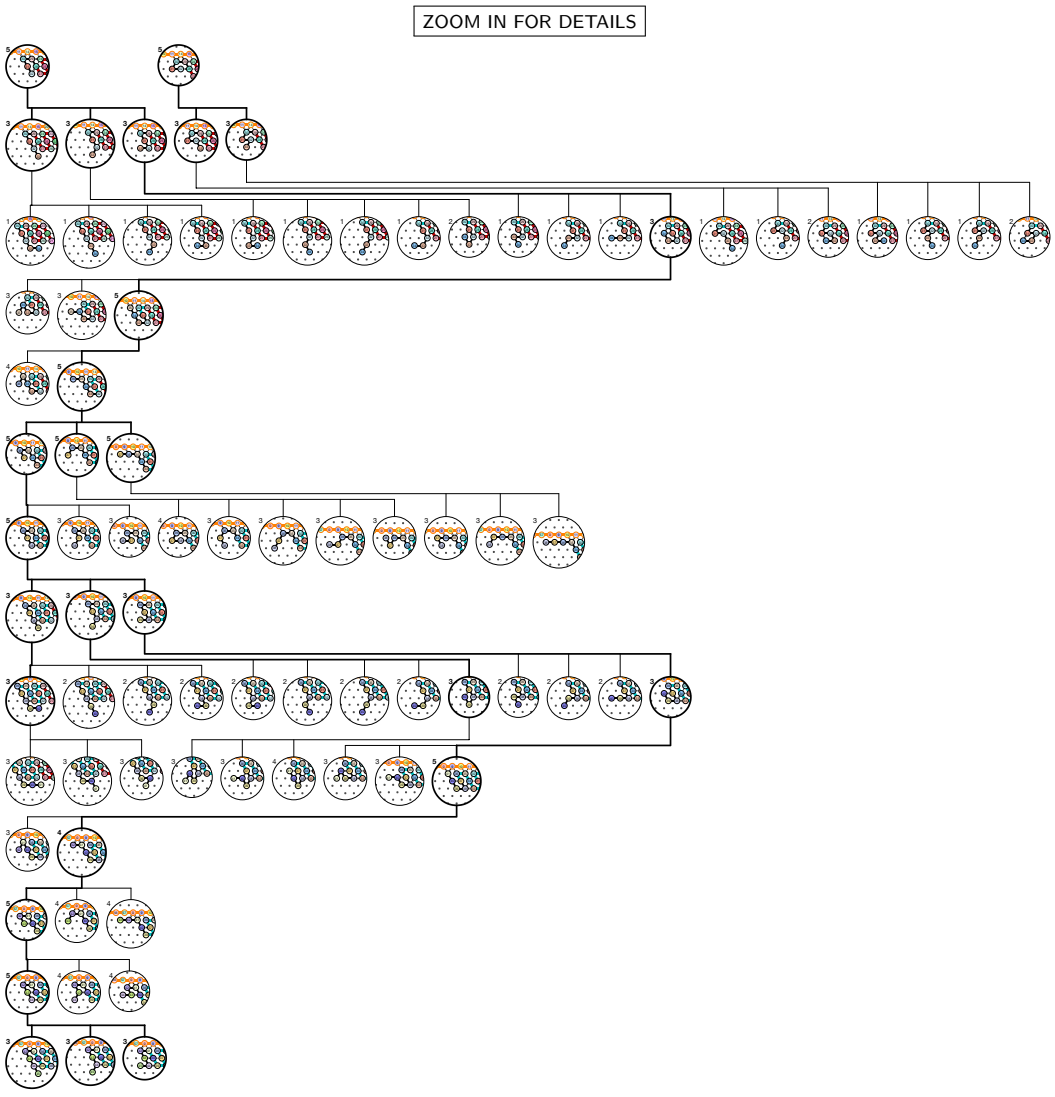
This appendix contains the certificates for most conformations of the half-adder modules. Others (conformations E, F, G, K) could not be included, due to their huge size. When reading these certificates, the top circles represent the initial conformations at the end of the previous module (there might be several of them). Then, the number of new bonds of each of the local conformations is written next to that conformation.

Some conformations (such as conformation D) initially start with several local conformations, but some of them are immediately trapped, in the sense that the last transcribed bead is inside a closed area of the plane, preventing the primary structure to grow further.

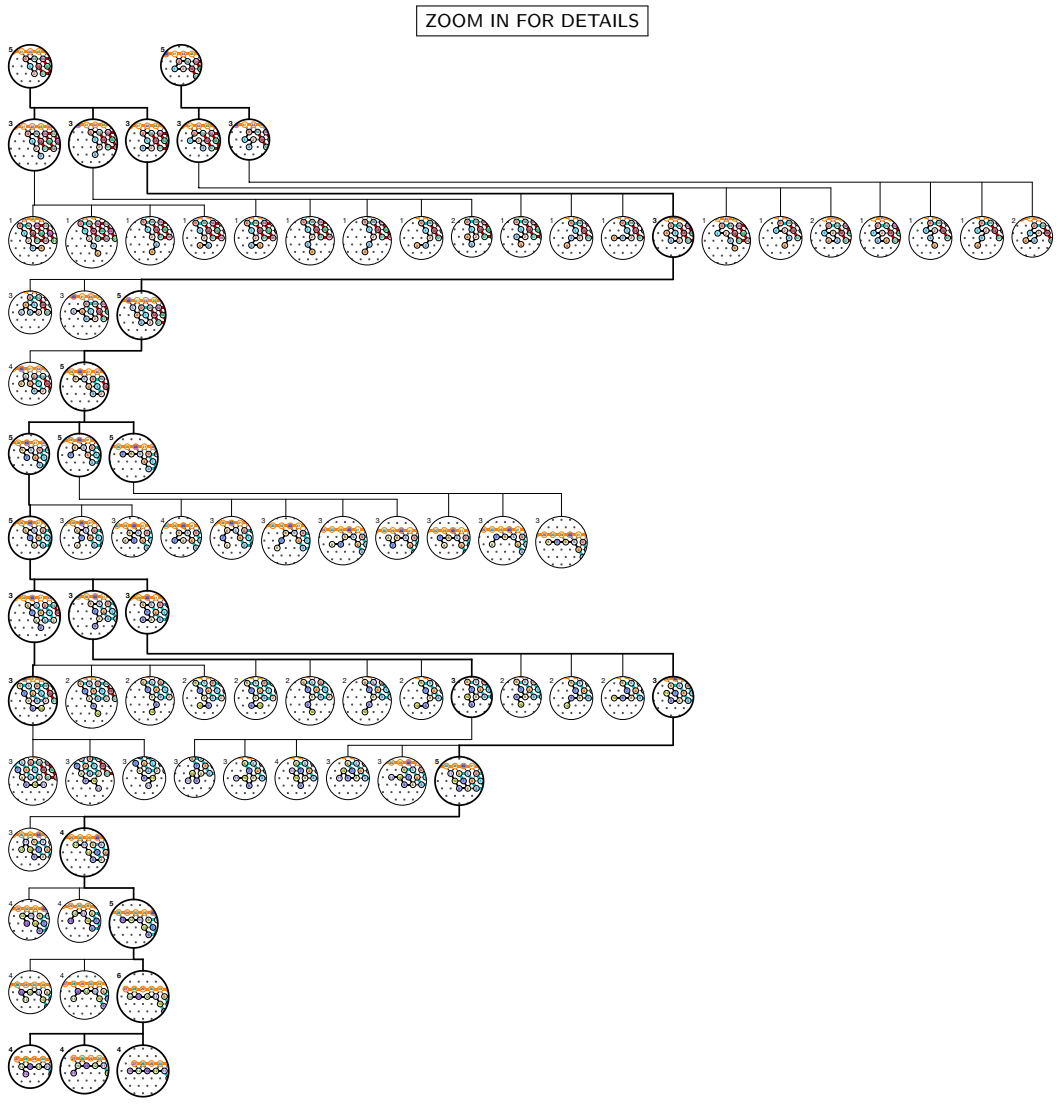
A.1 Conformation A



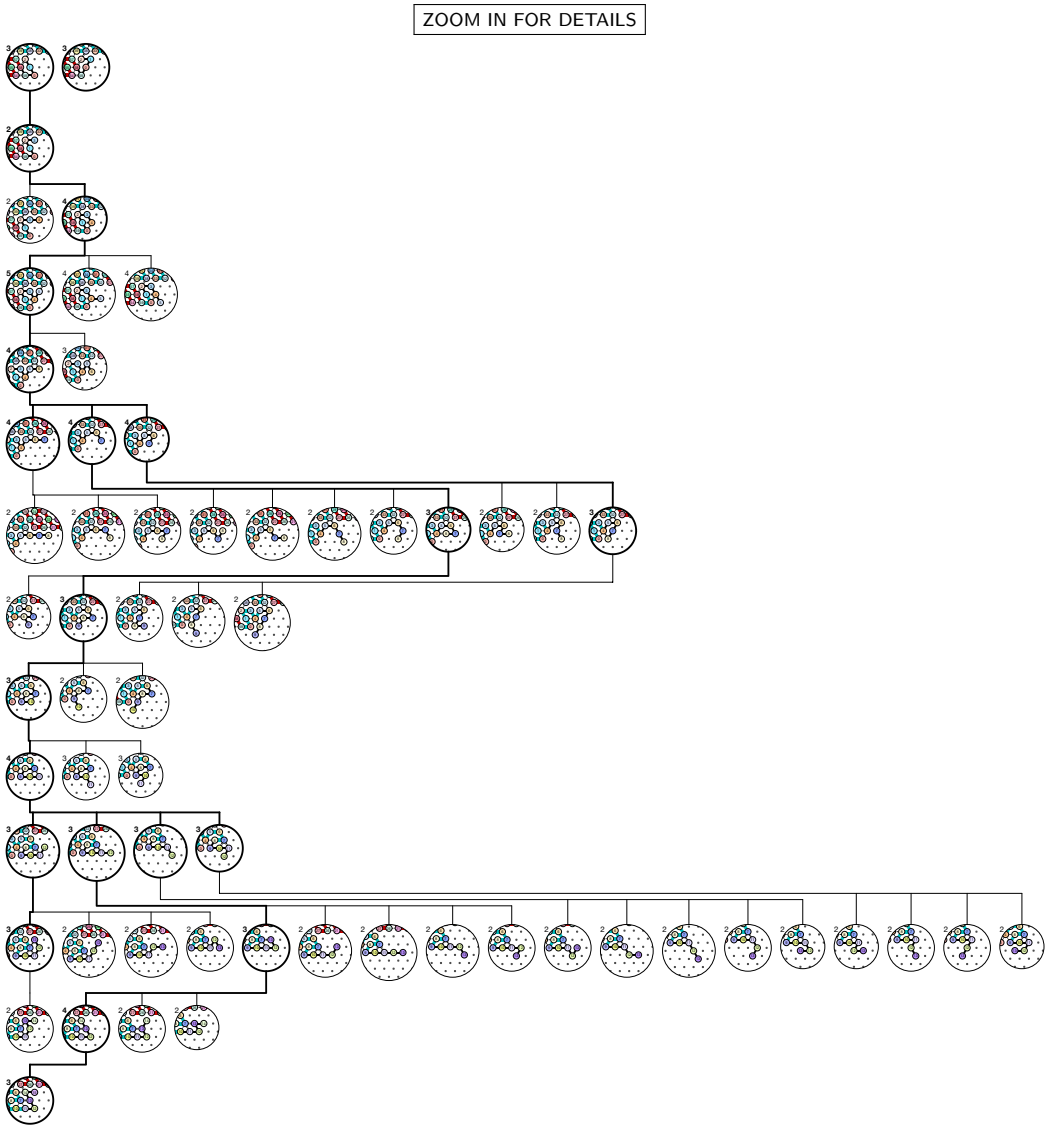
A.2 Folding certificate for Conformation B



A.3 Folding certificate for Conformation C

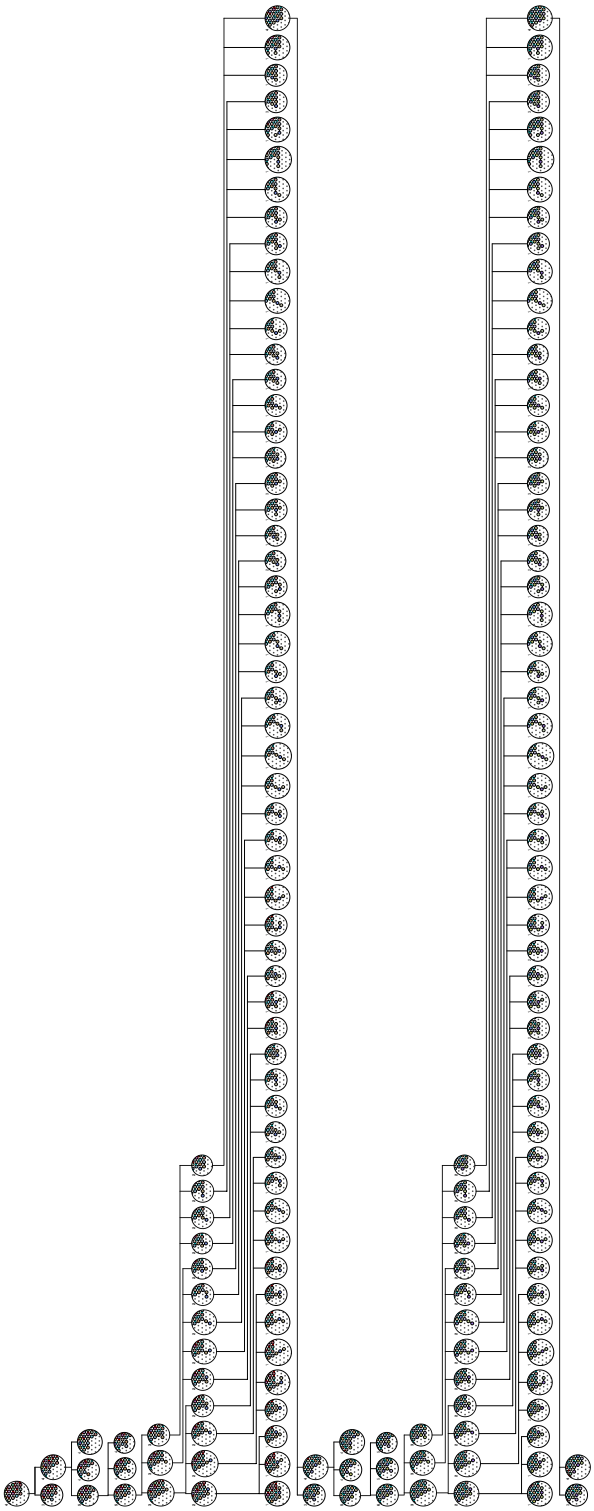


A.4 Folding certificate for Conformation D

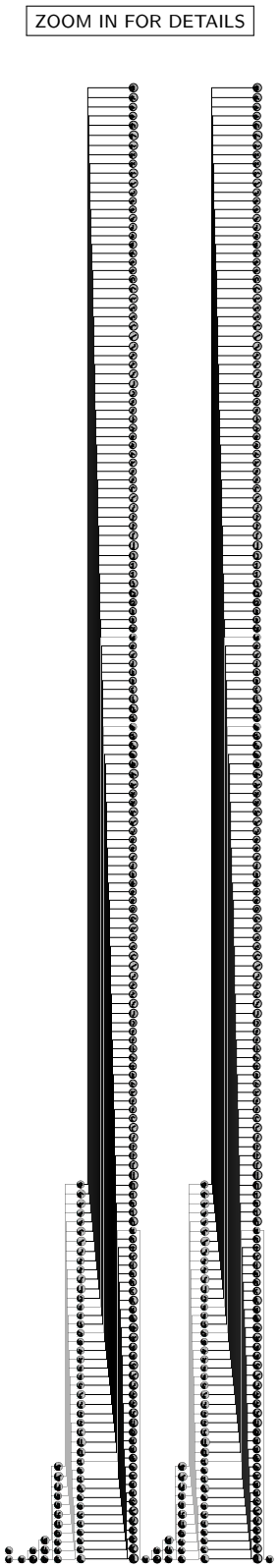


A.5 Folding certificate for Conformation E

ZOOM IN FOR DETAILS

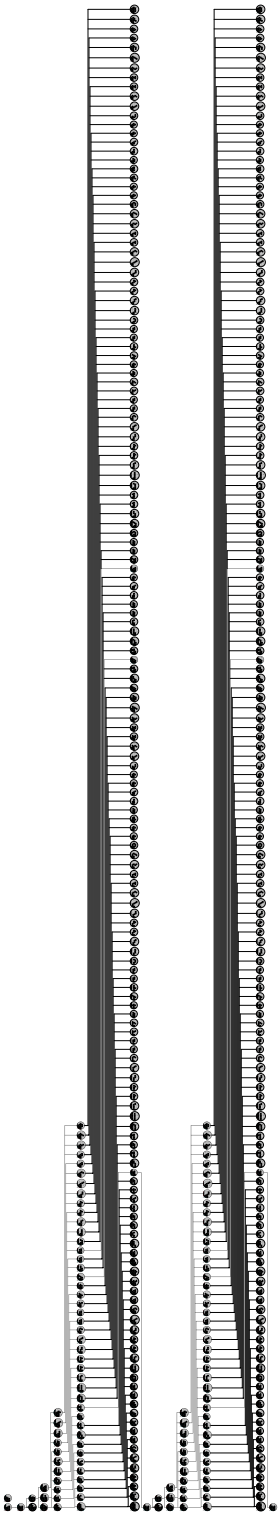


A.6 Folding certificate for Conformation F

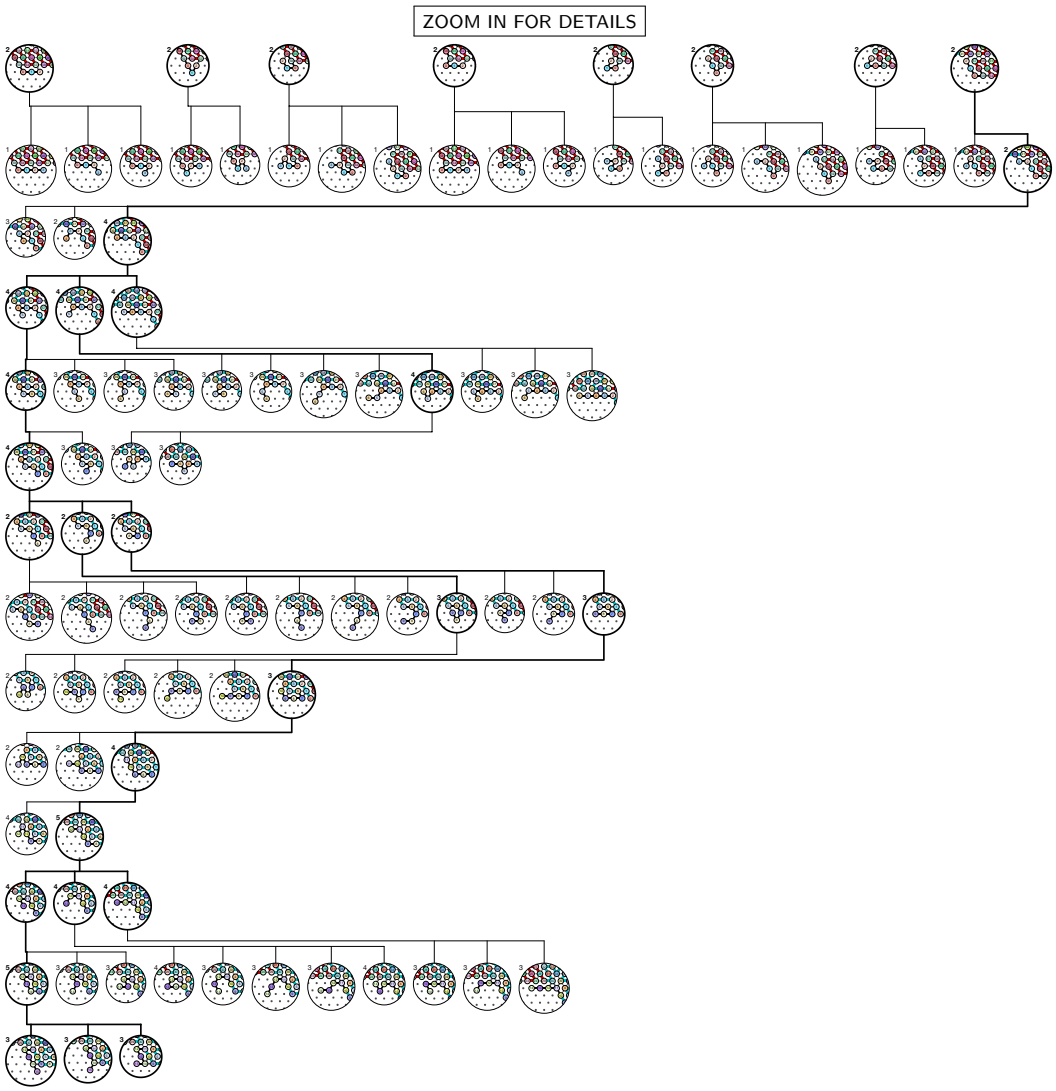


A.7 Folding certificate for Conformation G

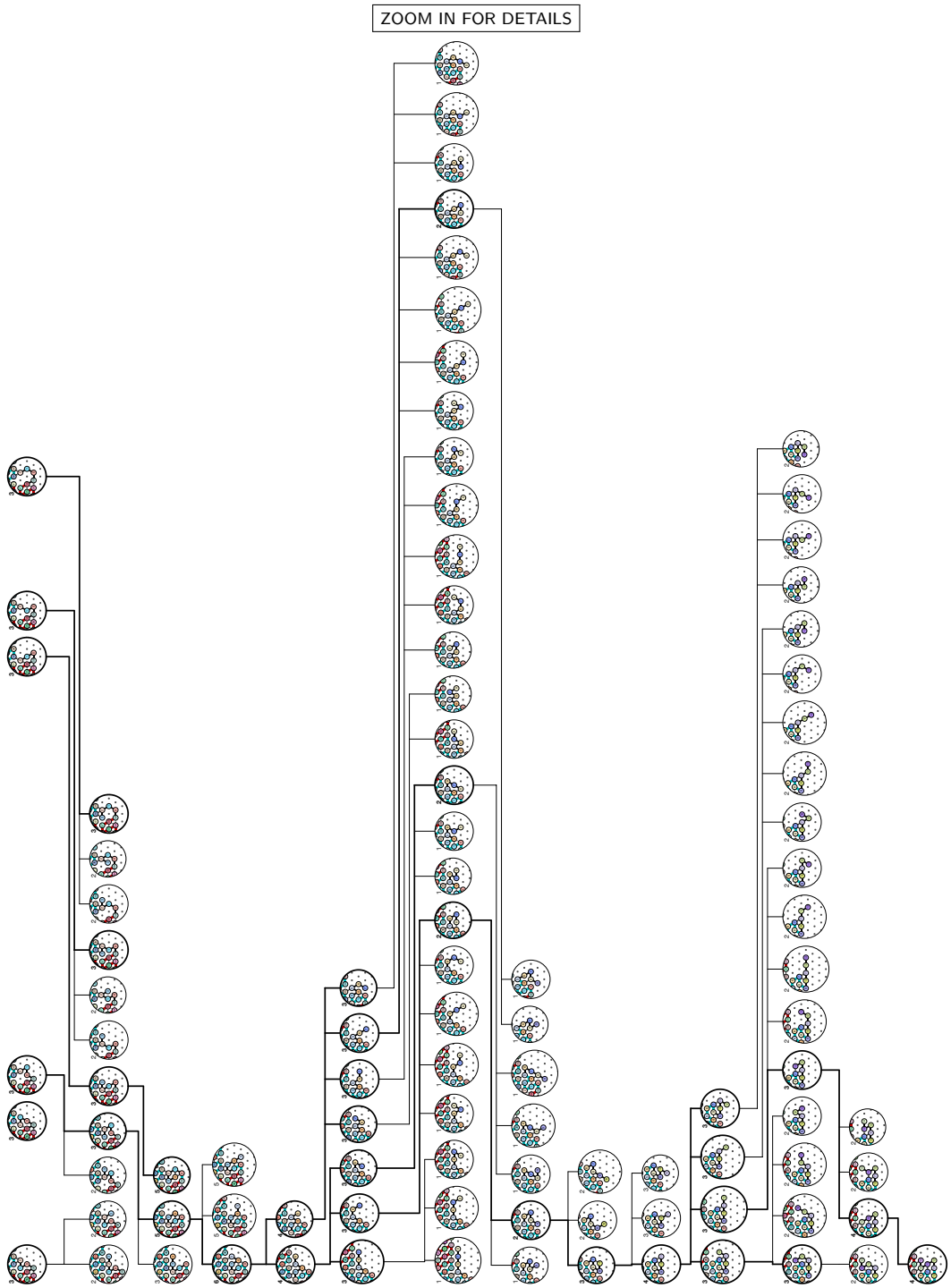
ZOOM IN FOR DETAILS



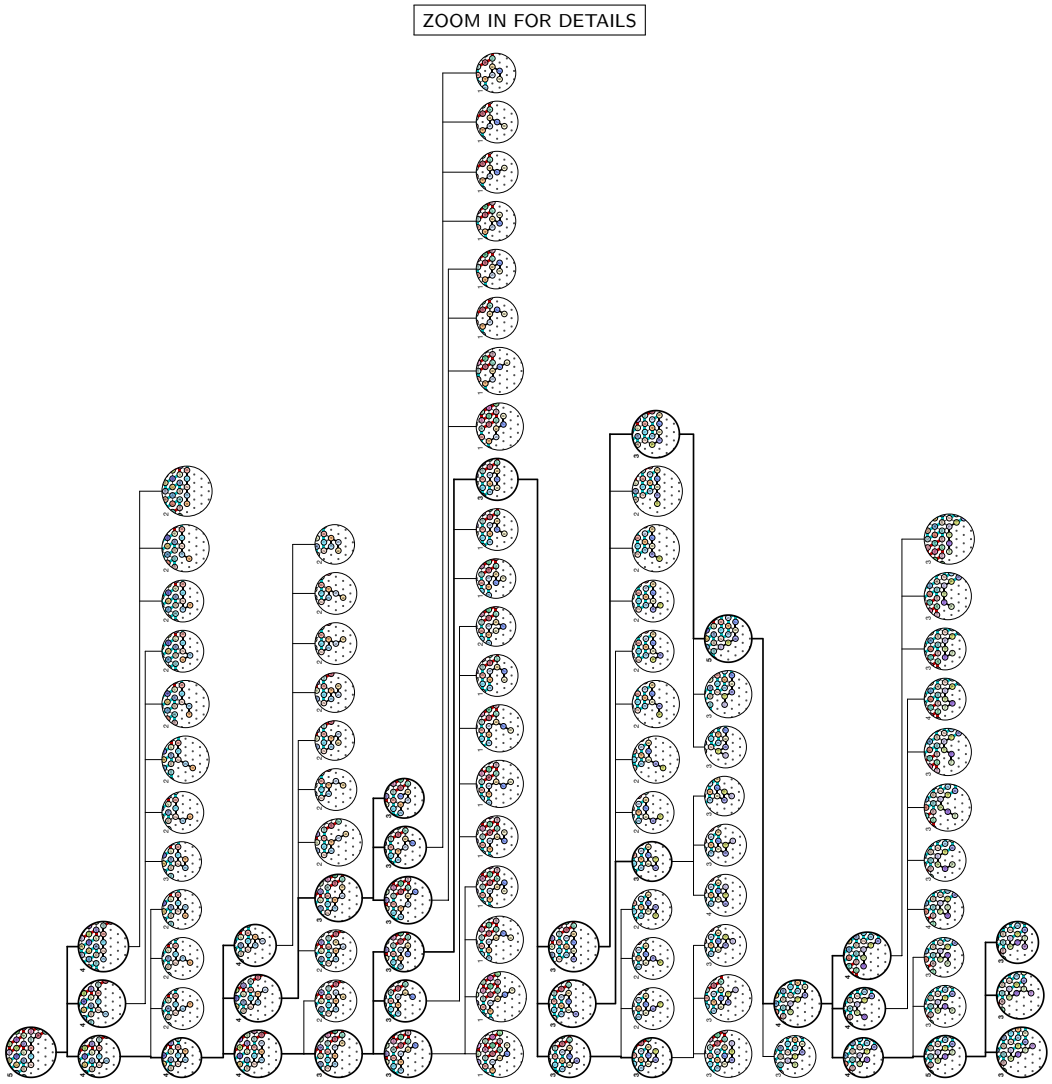
A.8 Folding certificate for Conformation H



A.9 Folding certificate for Conformation I

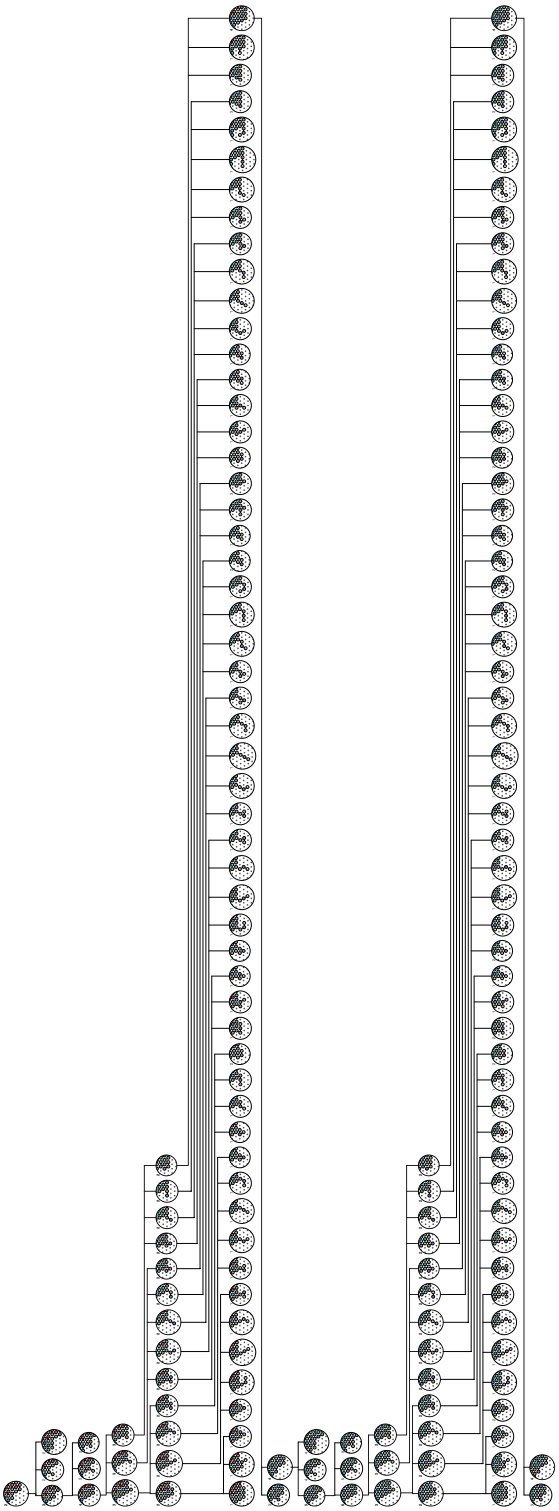


A.10 Folding certificate for Conformation J



A.11 Folding certificate for Conformation K

ZOOM IN FOR DETAILS



A.12 Folding certificate for Conformation L