Transcript Design Problem of Oritatami Systems

Yo-Sub Han[1], Hwee Kim[2](✉), and Shinnosuke Seki[3]

[1] Department of Computer Science, Yonsei University, 50 Yonsei-Ro, Seodaemun-Gu, Seoul 03722, Republic of Korea
emmous@yonsei.ac.kr

[2] Department of Mathematics and Statistics, University of South Florida, 12010 USF Cherry Drive, Tampa, FL 33620, USA
hweekim@mail.usf.edu

[3] Department of Computer and Network Engineering, University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 1828585, Japan
s.seki@uec.ac.jp

Abstract. RNA cotranscriptional folding refers to the phenomenon in which an RNA transcript folds upon itself while being synthesized out of a gene. Oritatami model is a computation model of this phenomenon, which lets its sequence (transcript) of beads (abstract molecules) fold cotranscriptionally by the interactions between beads according to its ruleset. We study the problem of designing a transcript that folds into the given conformation using the given ruleset, which is called the transcript design problem. We prove that the problem is computationally difficult to solve (NP-hard). Then we design efficient poly-time algorithms with additional restrictions on the oritatami system.

1 Introduction

A single-stranded RNA is synthesized sequentially from its DNA template by an RNA polymerase enzyme (*transcription*). The RNA transcript folds upon itself according to the base pairing rule—(A, U) and (C, G)—with respect to hydrogen bonds and gives rise to functional 3D-structures. Note that a synthesis direction and a rate at which nucleotides are added allow an RNA to fold over a predefined pathway into a non-equilibrium structure while being transcribed [14]. This phenomenon is called *cotranscriptional folding*.

Cotranscriptional folding plays an important role in algorithmic self-assembly. For example, Geary et al. [6] studied the architecture for RNA tiles (called RNA origami) and proposed a method to design a single-stranded RNA that cotranscriptionally folds into a target structure. Oritatami model (OM) is the first mathematical model for algorithmic self-assembly by cotranscriptional folding [5]. Given a sequence of molecules, OM assumes that the sequence is transcribed linearly, and predicts a geometric structure of the folding based on the reaction rate of the folding. An oritatami system (OS) in OM defines a sequence of beads (which is the transcript) and a set of rules for possible intermolecular reactions between beads. Here is how OS runs: Given a sequence of beads, the system takes a single bead (we call a current bead) together with a lookahead of a few succeeding beads, and determines the best location of the current bead that maximizes the number of possible interactions from a possible transcription of the lookahead. The lookahead represents the reaction rate of the cotranscriptional folding and the number of interactions represents the energy level. Researchers designed several OSs including a binary counter [4] and a Boolean formula simulator [9]. It is known that OM is Turing complete [5] and there are several methods to optimize OSs [8,10] (Fig. 1).

RNA Origami	Oritatami System
Nucleotides	Beads
Transcript	Sequence of beads connected by a line
h-bonds between nucleotides	Interactions
Cotranscriptional folding rate	Delay
Resulting secondary structure	Conformation

(a)

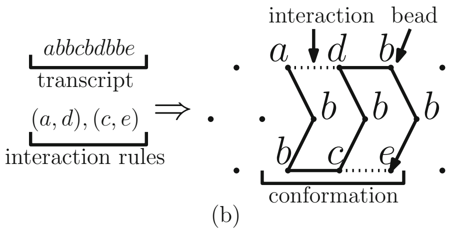

Fig. 1. (a) Analogy between RNA origami and oritatami system. (b) Visualization of oritatami system and its terms.

The inverse of RNA folding is RNA design: given a secondary structure, find a sequences of beads that uniquely folds into the input structure. If there are several possible foldings that the sequence can fold, then all the others must have less pairs than the input structure. We call this problem the RNA design problem. Hofacker et al. [12] introduced the RNA design problem and the complexity of the problem is still unknown [1]. The problem has applications in pharmaceutical research, biochemistry, synthetic biology or RNA nanostructures [2,7]. We consider the RNA design problem of an OS. In particular, we consider the case when we have the complete information about an OS including the bead type alphabet, pairing ruleset, delay, arity, and its final conformation except for beads on the conformation, and we need to find the transcript that folds the target conformation. Similar to the RNA design problem, this problem can be useful in several applications. For example, given a target structure and a generating system (OS), we can determine whether or not the generating system can produce the target structure and, if so, what is the correct transcript that indeed produces the target structure.

We first propose a general parameterized algorithm to solve the transcript design problem (TDP). We then tackle the CTDP, a restricted version of TDP where the ruleset is complementary. We prove that the CTDP is computationally difficult (NP-hard). Yet we also show that with a few restrictions on delay δ, arity α and the size $|\mathcal{H}|$ of the ruleset, we can solve the CTDP in linear time.

– CTDP is NP-hard (Theorem 2).

- CTDP is NP-complete when $\delta = 3$ and $|\mathcal{H}| = 3$ (Theorem 3).
- CTDP can be solved in linear time when $\delta = 1, |\mathcal{H}| = 1$, $\alpha = 1$ or $\alpha \geq 4$ (Lemmas 1 and 2).

2 Preliminaries

Let $w = a_1 a_2 \cdots a_n$ be a string over Σ for some integer n and bead types $a_1, \ldots, a_n \in \Sigma$. The *length* $|w|$ of w is n. For two indices i, j with $1 \leq i \leq j \leq n$, we let $w[i, j]$ be the substring $a_i a_{i+1} \cdots a_{j-1} a_j$; we use $w[i]$ to denote $w[i, i]$. We use w^n to denote the catenation of n copies of w.

Oritatami systems operate on the triangular lattice $\mathbb{T}$ with the vertex set V and the edge set E. A conformation instance, or *configuration*, is a triple (P, w, H) of a directed path P in $\mathbb{T}$, $w \in \Sigma^* \cup \Sigma^{\mathbb{N}}$, and a set $H \subseteq \{(i, j) \mid 1 \leq i, i + 2 \leq j, \{P[i], P[j]\} \in E\}$ of hydrogen-bond-based interactions (interactions for short). This is to be interpreted as the sequence w being folded while its i-th bead $w[i]$ is placed on the i-th point $P[i] \in V$ along the path and there is an interaction between the i-th and j-th beads if and only if $(i, j) \in H$. The fact that $i + 2 \leq j$ implies that $w[i]$ and $w[i+1]$ cannot form an interaction, since they are covalently bonded. Configurations (P_1, w_1, H_1) and (P_2, w_2, H_2) are *congruent* provided $w_1 = w_2$, $H_1 = H_2$, and P_1 can be transformed into P_2 by a combination of a translation, a reflection, and rotations by 60°. The set of all configurations congruent to a configuration (P, w, H) is called the *conformation* of the configuration and denoted by $C = [(P, w, H)]$. We call w a *primary structure* of C.

A ruleset $\mathcal{H} \subseteq \Sigma \times \Sigma$ is a symmetric relation specifying between which bead types can form an interaction. A ruleset is *complementary* if for all $a \in \Sigma$, there exists a unique $b \in \Sigma$ such that $(a, b) \in \mathcal{H}$. For a complementary ruleset, we denote the pairing bead type b as $\overline{a}$. An interaction $(i, j) \in H$ is *valid with respect to* $\mathcal{H}$, or simply $\mathcal{H}$*-valid*, if $(w[i], w[j]) \in \mathcal{H}$. We say that a conformation C is $\mathcal{H}$*-valid* if all of its interactions are $\mathcal{H}$-valid. For an integer $\alpha \geq 1$, C is *of arity* α if the maximum number of interactions per bead is α, that is, if for any $k \geq 1$, $\big|\{i \mid (i, k) \in H\}\big| + \big|\{j \mid (k, j) \in H\}\big| \leq \alpha$ and this inequality holds as an equation for some k. By $\mathcal{C}_{\leq\alpha}$, we denote the set of all conformations of arity at most α.

Oritatami systems grow conformations by elongating them under their own ruleset. For a finite conformation C_1, we say that a finite conformation C_2 is an *elongation* of C_1 by a bead $b \in \Sigma$ under a ruleset $\mathcal{H}$, written as $C_1 \xrightarrow{\mathcal{H}}_b C_2$, if there exists a configuration (P, w, H) of C_1 such that C_2 includes a configuration $(P \cdot p, w \cdot b, H \cup H')$, where $p \in V$ is a point not in P and $H' \subseteq \big\{(i, |P|{+}1) \bigm| 1 \leq i \leq |P| - 1, \{P[i], p\} \in E, (w[i], b) \in \mathcal{H}\big\}$. This operation is recursively extended to the elongation by a finite sequence of beads as follows: For any conformation C, $C \xrightarrow{\mathcal{H}}_\lambda C$; and for a finite sequence of beads w and a bead b, a conformation C_1 is elongated to a conformation C_2 by $w \cdot b$, written as $C_1 \xrightarrow{\mathcal{H}}_{w \cdot b} C_2$, if there is a conformation C' that satisfies $C_1 \xrightarrow{\mathcal{H}}_w C'$ and $C' \xrightarrow{\mathcal{H}}_b C_2$.

An *oritatami system* (OS) is a 6-tuple $\Xi = (\Sigma, w, \mathcal{H}, \delta, \alpha, C_\sigma = [(P_\sigma, w_\sigma, H_\sigma)])$, where $\mathcal{H}$ is a *ruleset*, $\delta \geq 1$ is a *delay*, and C_σ is an $\mathcal{H}$-valid initial *seed* conformation of arity at most α, upon which its *transcript* $w \in \Sigma^* \cup \Sigma^\omega$ is to be folded by stabilizing beads of w one at a time and minimize energy collaboratively with the succeeding $\delta - 1$ nascent beads. The energy of a conformation $C = [(P, w, H)]$ is $U(C) = -|H|$; namely, the more interactions a conformation has, the more stable it becomes. The set $\mathcal{F}(\Xi)$ of conformations *foldable* by this system is recursively defined as follows: The seed C_σ is in $\mathcal{F}(\Xi)$; and provided that an elongation C_i of C_σ by the prefix $w[1:i]$ be foldable (i.e., $C_0 = C_\sigma$), its further elongation C_{i+1} by the next bead $w[i{+}1]$ is foldable if

$$C_{i+1} \in \underset{\substack{C \in \mathcal{C}_{\leq\alpha} \text{s.t.} \\ C_i \xrightarrow{\mathcal{H}}_{w[i+1]} C}}{\operatorname{argmin}} \min \left\{ U(C') \;\middle|\; C \xrightarrow{\mathcal{H}}{}^{*}_{w[i+2:i+k]} C', k \leq \delta, C' \in \mathcal{C}_{\leq\alpha} \right\}. \quad (1)$$

Once we have C_{i+1}, we say that the bead $w[i{+}1]$ and its interactions are *stabilized* according to C_{i+1}. A conformation foldable by Ξ is *terminal* if none of its elongations is foldable by Ξ. An OS is *deterministic* if, for all i, there exists at most one C_{i+1} that satisfies (1). Namely, a deterministic OS folds into a unique terminal conformation.

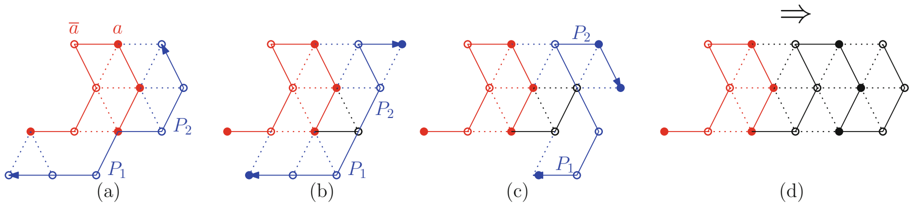

Fig. 2. An example OS with delay 3 and arity 4. Filled and unfilled circles represent bead types a and $\overline{a}$, respectively. The seed is colored in red, elongations are colored in blue, and the stabilized beads and interactions are colored in black. (Color figure online)

Figure 2 illustrates an example of an OS with delay 3, arity 4, complementary ruleset $\{(a, \overline{a})\}$ and transcript $w = \overline{a}\overline{a}\overline{a}aaa\overline{a}\overline{a}\overline{a}$; in (a), the system tries to stabilize the first bead $\overline{a}$ of the transcript, and the elongation P_1 gives 2 interactions, while the elongation P_2 gives 4 interactions, which is the most stable one. Thus, the first bead $\overline{a}$ is stabilized according to the location in P_2. In (b) and (c), P_2 is the most stable elongation and $\overline{a}$'s are stabilized according to P_2. As a result, the terminal conformation is given as in (d). Note that the system grows the terminal conformation straight without external interactions, and we can use an arbitrary prefix of $(\overline{a}\overline{a}\overline{a}aaa)^*$ to construct a conformation of an arbitrary length. This example is called a *glider* [4] and used in Sect. 3.1.

Conformations C_1 and C_2 are *isomorphic* if there exist an instance (P_1, w_1, H_1) of C_1 and an instance (P_2, w_2, H_2) of C_2 such that $P_1 = P_2$ and $H_1 = H_2$. For two sets $\mathcal{C}_1$ and $\mathcal{C}_2$ of conformations, we say that two sets are isomorphic if there exists a one-to-one mapping $C_1 \in \mathcal{C}_1 \rightarrow C_2 \in \mathcal{C}_2$ such that C_1 and C_2 are isomorphic. We say that two oritatami systems are isomorphic if they fold the isomorphic set of foldable terminal conformations.

We define the transcript design problem (TDP).

Problem 1 (Transcript Design Problem (TDP)). Given an alphabet Σ, a ruleset $\mathcal{H}$, a delay δ, an arity α, a seed $C_\sigma = [(P_\sigma, w_\sigma, H_\sigma)])$, a path P and a set H of interactions, find a transcript w such that an OS $\Xi = (\Sigma, w, \mathcal{H}, \delta, \alpha, C_\sigma)$ uniquely folds a terminal conformation $C = [(P, w, H)]$.[1]

The complementary transcript design problem (CTDP) is a subproblem of the TDP in which an input ruleset is required to be complementary.

3 Hardness of the TDP and the CTDP

We propose a generalized algorithm to solve the TDP, and prove hardness of CTDP. We first introduce the concept of the event horizon and its context, which will be used in the rest of the paper.

By definition, the stabilization of a bead $w[i]$ in a delay-δ OS is not affected by any bead whose distance from $w[i-1]$ is greater than $\delta+1$. On the triangular lattice, we may draw a hexagonal border of radius $\delta+1$ from $w[i-1]$ to denote the set of points that may affect the stabilization, and we call the hexagon the *event horizon* of $w[i]$. Note that the event horizon can have at most $3(\delta+1)(\delta+2)$ beads within, aside from $w[i]$. We call the already stabilized beads within the event horizon, along with interactions, as the *event horizon context* to represent the context used to stabilize $w[i]$. Thus, if two beads $w[i]$ and $w[j]$ have the same event horizon context, then $w[i]$ and $w[j]$ will be stabilized at the same position with the same interactions, considering a translation, a reflection or a rotation (see Fig. 3.).

Now, we define the dependence distance of a TDP instance.

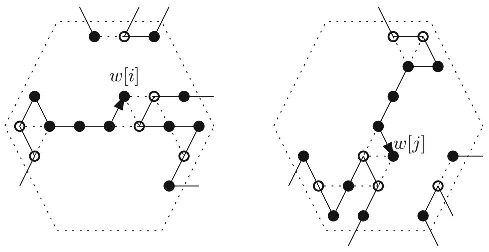

Fig. 3. Two same event horizon contexts when $\delta = 2$ and we have two bead types (black and white circles). The current bead, pointed by an arrow, is stabilized at the same position in both event horizon contexts.

[1] For the hardness proof, we use the decision variant of TDP, which determines whether or not such a transcript exists.

Definition 1. *Given a TDP instance* $(\Sigma, \mathcal{H}, \delta, \alpha, C_\sigma, P, H)$*, we define the dependence distance of the TDP instance as follows: Let* $w[i]$ *be the bead on the ith point of* P*. For each bead* $w[i]$*, let* r_i *be the smallest index such that while stabilizing* $w[i]$*,* $w[r_i]$ *is in the event horizon context of* $w[i]$*. We call* $\max(i+\delta-1-r_i)$ *the dependence distance.*

Namely, the dependence distance is the upper bound of the distance between a bead $w[i]$ and another bead $w[j]$ such that $w[j]$ affects the stabilization of $w[i]$. Note that the distance is independent from the delay of the system. Once the distance is bounded by a constant t, we can incrementally construct a transcript while having information of only t beads at a time, which results in the following theorem.

Theorem 1. *Given a TDP instance* $(\Sigma, \mathcal{H}, \delta, \alpha, C_\sigma, P, H)$*, we can solve the TDP in* $O(|\Sigma|^t \times |P|)$*, where* t *is the dependence distance of the TDP instance.*

Note that this general algorithm is fixed parameter linear. Next, we show that the CTDP is NP-hard in a general condition. We borrow the multi-chamber-gun construction from Ota and Seki [13] to reduce 1-IN-3-SAT to the CTDP at a long delay. The seed of multi-chamber-gun shape encodes the clauses of a given 1-IN-3-SAT instance. In order to go through the cannon tube as specified by the target conformation, the transcript must encode a satisfying assignment of truth values (`T/F`) to variables $v_1, v_2, \ldots, v_k$ for each of the m clauses in a uniform format like $(x_{1,1}x_{1,2}\ldots x_{1,k})(x_{2,1}\ldots x_{2,k})\ldots(x_{m,1}\ldots x_{m,k})$. For all $1 \leq i \leq k$, the assignments to v_i for every pair of the adjacent clauses are forced to be identical by chambers. The 1-IN-3-SAT instance is thus reduced to a TDP instance, and in fact, this reduction works with complementary ruleset.

Theorem 2. *For all* $\alpha \geq 1$*, the complementary transcript design problem (CTDP) at arity* α *is NP-hard. It remains NP-hard even if an input ruleset is restricted to be of size at most 2.*

3.1 Graph-Theoretic Approach to the CTDP

In the CTDP, since the ruleset is complementary, we may say that each bead type belongs to a rule in the ruleset. When the path P and the set H of interactions are given, we can retrieve necessary dependence conditions between two adjacent beads according to three different cases:

1. If two beads are connected with an interaction: Two beads should belong to the same rule.
2. If two beads are connected with a path: There is no necessary condition between two beads.
3. If there is no relationship between two beads: Two beads should not belong to the same rule, or two beads should have the same type.

We call these conditions static dependence (s-dependence in short), since these conditions are derived from the given path and the set of interactions, which do not include dynamics of stabilization of beads. From the first condition, if one of two beads is already stabilized or in the seed, we can find the bead type for the other bead. Moreover, if a set of beads are connected with interactions, one bead in the set determines bead types for the rest in the set. Therefore, we may regard this set of beads as a dependent set of beads. Each set should have one representative bead that represents the bead type assignment for all beads in the set, and additional information to find the transcript can be represented by the relationship between these representative beads. It takes $O(|w|)$ time to retrieve dependent sets from the given path and the set of interactions. When there exists a odd length cycle of interactions, we can immediately tell that the answer to the CTDP is no. Aside from this case, since each dependent set uses bead types that belong to one rule, we may represent each rule by a distinct color and regard the CTDP as a variant of the graph coloring problem (Fig. 4).

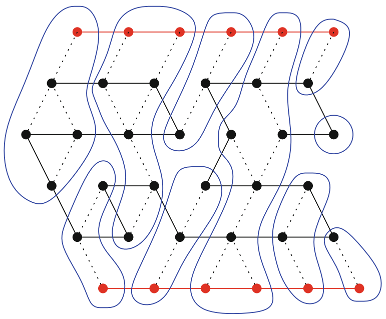

Fig. 4. Finding dependent sets. The seed is colored in red, and the dependent sets are colored in blue. (Color figure online)

There exists another category of conditions called dynamic dependence (d-dependence in short), which include dynamics of stabilization of beads. While stabilizing each bead of the transcript, there should exist one elongation of length δ that is used to stabilize the current bead at the designated point. Also, for all elongations that are not used to stabilize the current bead at the designated point, the number of interactions should be less than the number of interactions from the most stable elongation. For each possible bead type assignment for beads within the event horizon context, we can determine the possible bead type assignment for the current bead. According to dynamic dependence, there may exist some dependent sets that should have interactions with each other, and thus can be merged.

Now, we prove that the CTDP is NP-complete even for delay 3.

Theorem 3. *The CTDP is NP-complete when* $\delta = 3$ *and* $|\mathcal{H}| = 3$.

Proof. Once a proper transcript is given, we can check whether the given transcript successfully folds along the given path with the given set of interactions within $O(|w|)$ time. Thus, the problem is NP.

We prove that the problem is NP-hard, using the reduction from the planar 3-coloring problem [3]. Suppose that we are given a planar graph with n vertices. We can embed the graph on a square grid graph of size $O(n^2)$ [11]. An edge in the original planar graph is represented by a set of vertical and horizontal edges on the square grid graph.

The basic idea is to construct a path that spans the square grid graph horizontally using zigs and zags. We will represent a vertex from the original planar graph by a dependent set of beads connected with interactions, and an edge by a boundary between two dependent sets. We will force the adjacent dependent sets assign bead types from different rules.

We use the glider in Fig. 2 as a basic module, since it uses only 2 complementary bead types. We assume that we start to span the square grid graph from the northeast corner. We combine 24 beads in one zig and adjacent zag as one module to represent one vertex of the square grid as in Fig. 5. Note that all vertices are connected with interactions. The same paths are used to represent a horizontal edge of the square grid, and a vertical edge of the square grid is represented by interactions between two modules.

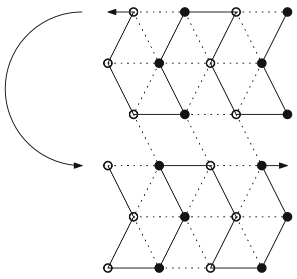

Fig. 5. A module that represent a vertex of the square grid

First, we present the module for a vertical edge of the square grid. If the edge does not represent an edge from the original graph, then the upper vertex module and the lower vertex module should be connected by interactions as in Fig. 6(a). If the edge represents an edge from the original graph, bead types from different rules should be assigned for the upper vertex module and the lower vertex module respectively. Thus, there should be no interaction between the upper vertex module and the lower vertex module, as in Fig. 6(b). In the red circle, a bead in the lower vertex module has no interaction with both complementary bead types in the upper vertex module, and they are not connected by the path either. This forces the assignment of bead types from different rules for the upper vertex module and the lower vertex module respectively.

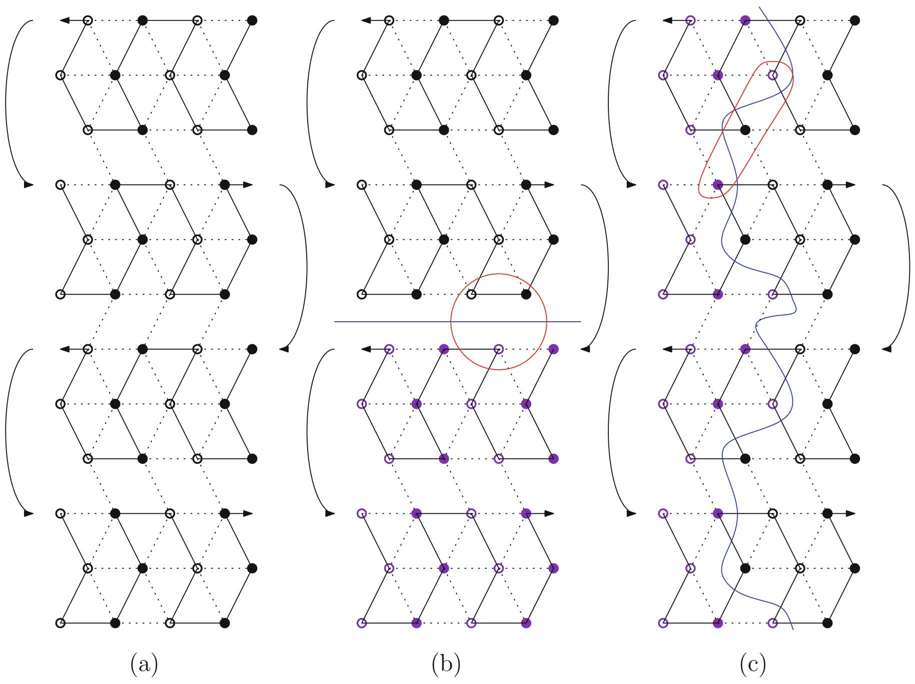

Fig. 6. (a) The module that represents the lack of a vertical edge. (b) The module representing the presence of a vertical edge. (c) The module representing the presence of a horizontal edge.

Next, we present the module for a horizontal edge of the square grid. If the edge does not represent an edge from the original graph, we can use the same module as the vertex module. If the edge represents an edge from the original graph, we need to embed two horizontally dependent sets in the module. Figure 6(c) shows the module for two horizontal edges of the square grid, where the blue line is the borderline between two dependent sets. While folding in a glider path, the module successfully embeds two dependent sets. In the red line, a bead in the right dependent set has no interaction with both complementary bead types in the left dependent set, and they are not connected by the path either. This forces the assignment of bead types from different rules for two dependent sets.

Lastly, we present the module for turns of zigs and zags, which should also represent a vertical edge of the square grid. If the edge does not represent an edge from the original graph, then the upper vertex module and the lower vertex module should be connected by interactions as in Fig. 7(a). If the edge represents an edge from the original graph, there should be no interaction between the upper vertex module and the lower vertex module, as in Fig. 7(b). In the red circle, a bead in the lower vertex module has no interaction with both complementary bead types in the upper vertex module, and they are not connected by the path either. This forces the assignment of bead types from different rules for the upper vertex module and the lower vertex module respectively.

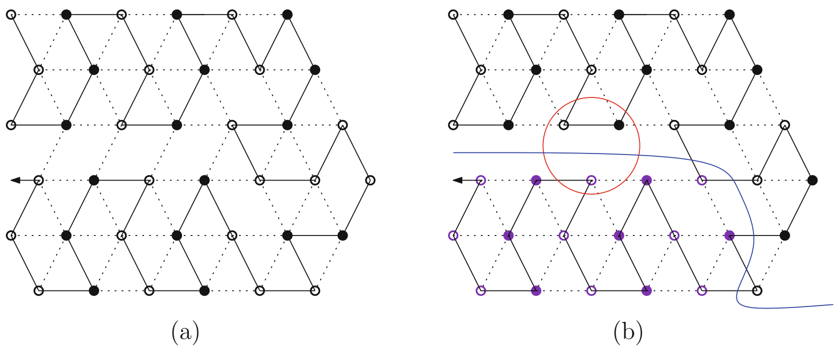
Fig. 7. The module for a turn. (a) The module does not represent a vertical edge. (b) The module represents a vertical edge.

We have successfully transformed a vertex in the original graph to a dependent set of beads, and an edge to a boundary between adjacent dependent sets, and forced that adjacent dependent sets should have bead types from different rules. Thus, we can color the original graph with three colors if and only if we can find a bead type assignment that satisfies s-dependence using three complementary rules. Moreover, for all cases, if s-dependence in a module is satisfied, so is d-dependence—regardless of the possible context, the module folds as a desired conformation. Thus, this bead type assignment implies a transcript that can be an answer for the reduced CTDP instance. □

4 Delay-1 CTDP

Knowing that the TCP and the CTDP are NP-hard, we now try to find sufficient conditions that make the CTDP solvable in polynomial time. Here, we focus on the case where $\delta = 1$. Delay-1 CTDP is essentially different from the general CTDP. In the general CTDP, while stabilizing a bead, interactions in the most stable elongation may not appear in the terminal conformation if they are not from the current bead. Such interactions are called *phantom* interactions. However, when $\delta = 1$, there is no phantom interaction and we can explicitly count the number of interactions that are needed to stabilize each bead—the number of interactions that the current bead has in H. This explicit information helps us determine bead type relationships resulting from d-dependence, and design linear time algorithms to solve the CTDP under specific conditions.

Lemma 1. *We can solve the CTDP in $O(|w|)$ time when $\delta = 1$, $|\mathcal{H}| = 1$ and $\alpha \geq 4$.*

Proof. We start from writing s-dependence conditions between two adjacent beads in Sect. 3.1 when $|\mathcal{H}| = 1$.

1. If two beads are connected with an interaction: Two beads are of different types.
2. If two beads are connected with a path: There is no necessary condition between two beads.
3. If there is no relationship between two beads: Two beads have the same type.

Note that both the first and the third conditions uniquely determine the bead type of one based on the other.

If the delay of the system is 1, for each bead b_1 to stabilize, there are two different cases (See Fig. 8):

1. Stabilization by interactions: The bead is stabilized deterministically by at least one interaction with neighbors on the conformation. In this case, the bead may be stabilized at another point without these interactions.
2. Stabilization by geometry: The bead is stabilized deterministically by geometric constraints. In this case, the possible interactions that the current bead may have do not change the stabilization point.

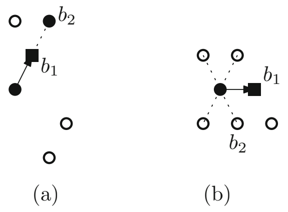

Fig. 8. Two cases when $\alpha \geq 4$. We assume that there are two types of beads: a black circle and a white circle. The current bead to stabilize is represented by a black square. (a) Stabilization by interactions (b) stabilization by geometry

In both cases, while stabilizing the bead b_1, the bead should have at least one already stabilized bead b_2, where two beads are connected with an interaction (the first condition of s-dependence) or there is no relationship between them (the third condition of s-dependence). Otherwise, the system becomes nondeterministic or the bead cannot stabilize at the designated point. Since $\alpha \geq 4$, b_2 can have up to 4 interactions aside from two neighboring beads on the path, and if $(b_1, b_2) \in \mathcal{H}$, b_1 and b_2 always have an interaction.

The first and the third conditions of s-dependence make the bead type assignment unique if the bead type of one of two beads is fixed. Therefore, for each bead, there exists unique bead type assignment resulting from the first or the third condition with an adjacent (already known) bead. Moreover, in both cases, since we are aware of all beads within the event horizon context, we can check that d-dependences are satisfied online: in other words, whether the current bead is stabilized as desired or not. Thus, the total runtime to find a transcript is $O(|w|)$.

Lemma 2. *We can solve the CTDP in $O(|w|)$ time when $\delta = 1$, $|\mathcal{H}| = 1$ and $\alpha = 1$.*

Proof. When $\alpha = 1$, once a bead forms an interaction with another, these two beads become inactive and cannot form an interaction anymore. We call beads that are not binded as active beads. For each bead b_1 to stabilize, there are three different cases (See Fig. 9.):

1. Stabilization by an interaction: The bead is stabilized deterministically by exactly one interaction with a neighbor on the conformation.
2. Stabilization by geometry, having an active neighbor: The bead is stabilized deterministically by geometric constraints. In addition, there exists at least one neighboring bead which did not have an interaction so far, which we call an active neighbor.
3. Stabilization by geometry, not having an active neighbor: The bead is stabilized deterministically by geometric constraints. In addition, all neighboring beads have interactions already.

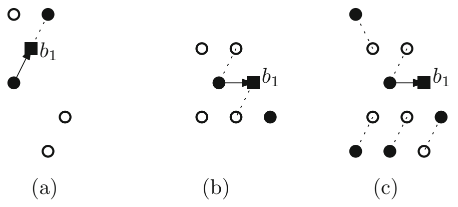

Fig. 9. Three cases when $\alpha = 1$. We assume that there are two types of beads: a black circle and a white circle. The current bead to stabilize is represented by a black square. (a) Stabilization by an interaction (b) stabilization by geometry, having an active neighbor (c) Stabilization by geometry, not having an active neighbor

We propose an algorithm to assign a bead type for these three cases.

1. Stabilization by geometry, not having an active neighbor: Since there is no active neighbor, we may assign an arbitrary bead type to the current bead at this timestamp. Thus, we introduce a new bead type variable v_{i+1}, given the most recent bead type variable v_i, and assign the bead type variable to the current bead.
2. Stabilization by geometry, having an active neighbor: Similar to the second case when $\alpha \geq 4$, we have a set of active neighbors whose bead types (or variables) are fixed. Based on the apparent interactions, we can assign the unique bead type (or variable) to the current bead, and may fix the bead type for a variable or merge two variables based on relationships within the event horizon context.

3. Stabilization by an interaction: Since the arity is 1, the current bead should have an active neighbor with the complementary bead type (or variable). Moreover, all active neighbors of neighbors of the previous bead except the stabilization point should have the same bead type as the current bead (or variable). Thus, we can assign the unique bead type (or variable) to the current bead, and may fix the bead type for a variable or merge two variables based on relationships within the event horizon context.

Note that for all cases, there exists an unique bead type (or variable) assignment for the current bead. Similar to the $\alpha \geq 4$ case in Lemma 1, we can check d-dependences are satisfied online. Moreover, possible changes on the variables (fixing the bead type or merging two variables) while stabilizing future beads do not change d-dependences and still result in the same isomorphic conformation. Thus, once we assign bead types (or variables) to the end of the transcript, we may assign arbitrary bead types for variables, and the resulting transcript always folds the conformation isomorphic to the original one. The total runtime to find a transcript is $O(|w|)$. □

Here, we relieve the CTDP by allowing isomorphism for the seed. Based on the relaxation, we claim that we may reduce the size of the ruleset without changing solvability, where the upper bound of the size of the ruleset is 27.

Lemma 3. *Let $P_1 = (\Sigma, \mathcal{H}, 1, 1, C_\sigma, P, H)$ be an instance of CTDP at delay 1 and arity 1. If $|\mathcal{H}| > 27$, one can construct a ruleset $\mathcal{H}' \subseteq \mathcal{H}$ of size 27 and the seed C'_σ over $\Sigma(\mathcal{H}')$ isomorphic to C_σ, such that if P_1 has a solution, then the instance of CTDP $P_2 = (\Sigma(\mathcal{H}'), \mathcal{H}', 1, 1, C'_\sigma, P, H)$ does.*

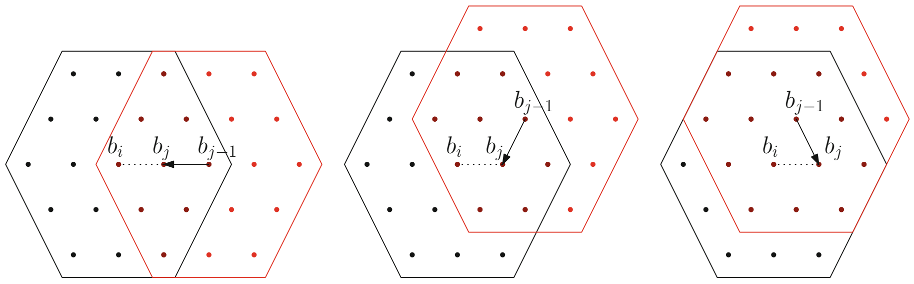

Fig. 10. The region of the influence of b_i at delay 1 and arity 1 in all the possible three cases modulo the reflectional symmetry along the line b_i–b_j. (Color figure online)

Proof. We claim that the bead type of a bead is dependent upon at most 26 other beads. Assume that the given seed C_σ consists of m beads and the path P consists of n beads. We index the beads of C_σ as $b_{-m+1}, b_{-m+2}, \ldots, b_{-1}, b_0$, where b_{-1} is connected to the first bead of P. For convenience, we also index the beads on P as $b_1, b_2, \ldots, b_n$, where $b_i = w[i]$.

We consider the relationship between two beads b_i and b_j, where $i < j$ and b_i and b_j have an interaction with each other. Since $\alpha = 1$, the preceding bead b_i must remain active when it is stabilized. For that, b_i may be a part of the seed C_σ, or there was only one empty neighbor of its predecessor b_{i-1} so that b_i was forced to be stabilized without interactions (Third case of the proof for Lemma 2). In the latter case, two of the neighboring beads of b_{i-1} can affect the stabilization of b_i. The bead b_i can affect the stabilization of another bead b_k for any $i + 1 \leq k < j$. In order for b_k to be affected by b_i, its predecessor b_{k-1} must have been stabilized in the event horizon context of b_{i+1} (The black hexagons in Fig. 10). The event horizon context has 19 points, 3 of which are to be stabilized by b_i, b_j, and b_{j-1}. Note that the two beads that can affect the stabilization of b_i are also in this event horizon context. Therefore, there can be at most 16 beads which can affect the stabilization of b_i or whose stabilization can be affected by b_i. The bead b_j is affected by at most 16 beads other than b_i, which are inside the event horizon context of b_j (The red hexagons in Fig. 10).

Now we have at most 32 beads that can be affected by b_i or affect b_j, but we may reduce the number by geometric constraints. Suppose we see all the neighbors of b_{j-1} except b_j. A bead at one of these neighbors, say p, if any, prevents a bead at the other side of p from b_{j-1} from affecting b_j. The number of beads that can affect b_j, denoted by $d(b_j)$, is thus at most 11. We can bound the number of beads that can affect b_i or be affected by b_i, which we denote by $d(b_i)$, by 15. The successor b_{i+1} of b_i is to be stabilized at one of the neighbors of b_i but the one for b_j. Being thus stabilized at a neighbor, say p', b_{i+1} geometrically prevents b_k from being affected by b_i if its predecessor b_{k-1} is stabilized at the other side of p from b_i. We call $d(b_i) + d(b_j)$ the *degree of dependence of the pair* (b_i, b_j). Then the *degree of dependence of* C_σ is the maximum of the degree of dependence of a pair (b_i, b_j) such that b_i is included in C_σ but b_j is not[2].

We have proved that the degree of dependence of C_σ is at most 26. It is well known that we can color a graph with $d + 1$ colors, where d is the maximum degree of a vertex. Here, we may regard each rule as a color. For each pair of beads, we may consider the degree of dependence and assign bead types from different rules for beads that are dependent to the pair. Thus, it is sufficient to have the ruleset of size 27 to color the transcript. □

If a CTDP instance has no answer, we may increase the size of the ruleset and use additional bead types to find an answer. Note that there exists a CTDP instance without an answer, regardless of the size of the ruleset, as in Fig. 11. Aside from these apparent contradictory cases, we prove that there is no lower bound for the size of the ruleset where we can always find a transcript for the CTDP.

Lemma 4. *Given* $n \geq 1$, *there exists a CTDP instance* $P_1 = (\Sigma, \mathcal{H}, 1, 3, C_\sigma, P, H)$ *with* $|\mathcal{H}| = n$ *such that there is no answer for* P_1, *but*

[2] This definition does not consider any pair both of whose beads are included in C_σ because such a pair is already inert at the beginning of folding.

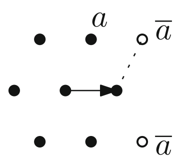

Fig. 11. One case where there is no answer for a CTDP instance, regardless of the size of the ruleset. The bead $w[1]$ both has and does not have an interaction with $\overline{a}$, which is a contradiction.

there exists a ruleset $\mathcal{H}' \supseteq \mathcal{H}$ *of size* $n+1$ *where the CTDP instance* $P_2 = (\Sigma, \mathcal{H}', 1, 3, C_\sigma, P, H)$ *has an answer.*

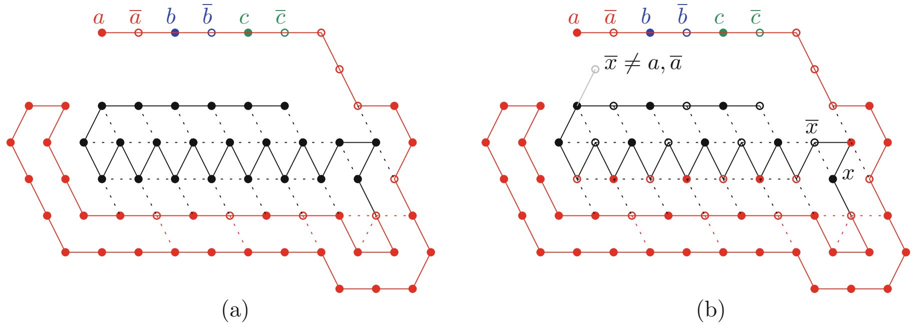
a $\overline{a}$ b $\overline{b}$ c $\overline{c}$
(a)
a $\overline{a}$ b $\overline{b}$ c $\overline{c}$
$\overline{x} \neq a, \overline{a}$
$\overline{x}$
x
(b)

Fig. 12. (a) A CTDP instance with $n = 3$. (b) Bead type assignment and constraints for $\overline{x}$

Proof. Fig. 12 (a) shows a CTDP instance that satisfies the lemma when $n = 3$ and $\mathcal{H} = \{(a, \overline{a}), (b, \overline{b}), (c, \overline{c})\}$. The red line is a seed, bead types in different rules are colored differently, and complementary bead types are represented by full and empty circles.

The first bead of the system is stabilized by geometry, and since neighboring a and $\overline{a}$ are active, the first bead should have a different type from both a and $\overline{a}$. Let us use the variable x to represent that bead type. Following the s-dependences, we can assign bead types as in Fig. 12(b).

Now, we consider d-dependences for a straight line of beads at the last part of the transcript. While stabilizing the first $\overline{x}$ on the line, which is denoted by a black empty circle, the bead is stabilized by one interaction with x. However, it may stabilize upper left if it can interact with either a or $\overline{a}$. Thus, $\overline{x}$ cannot be neither a or $\overline{a}$. The same analysis holds for the following $\overline{x}$'s, which result in that $\overline{x}$ should be different with all beads in the alphabet. This contradiction can be solved if we add a new rule $(d, \overline{d})$ and assign $x = d$. This CTDP instance can be extended for arbitrary n, and the lemma holds. □

Acknowledgements. Han was supported by NRF 2017K2A9A2A08000270 and NRF 2015R1D1A1A01060097, Kim was supported in part by the NIH grant R01 GM109459, and S. S. is supported in part by JST Program to Disseminate Tenure Tracking System, MEXT, Japan, No. 6F36, JSPS Grant-in-Aid for Young Scientists (A) No. 16H05854, and JSPS Bilateral Program No. YB29004.

References

1. Bonnet, É., Rzazewski, P., Sikora, F.: Designing RNA secondary structures is hard. In: Research in Computational Molecular Biology - 22nd Annual International Conference, RECOMB 2018 (2018, accepted)
2. Churkin, A., Retwitzer, M.D., Reinharz, V., Ponty, Y., Waldispühl, J., Barash, D.: Design of RNAs: comparing programs for inverse RNA folding. Brief. Bioinform. **19**, 350–358 (2017)
3. Garey, M.R., Johnson, D.S.: Computer and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, New York (1979)
4. Geary, C., Meunier, P., Schabanel, N., Seki, S.: Efficient universal computation by greedy molecular folding. CoRR, abs/1508.00510 (2015)
5. Geary, C., Meunier, P., Schabanel, N., Seki, S.: Programming biomolecules that fold greedily during transcription. In: Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science, pp. 43:1–43:14 (2016)
6. Geary, C., Rothemund, P.W.K., Andersen, E.S.: A single-stranded architecture for cotranscriptional folding of RNA nanostructures. Science **345**, 799–804 (2014)
7. Hales, J., Héliou, A., Manuch, J., Ponty, Y., Stacho, L.: Combinatorial RNA design: designability and structure-approximating algorithm in Watson-Crick and Nussinov-Jacobson energy models. Algorithmica **79**(3), 835–856 (2017)
8. Han, Y.-S., Kim, H.: Ruleset optimization on isomorphic oritatami systems. In: Brijder, R., Qian, L. (eds.) DNA 2017. LNCS, vol. 10467, pp. 33–45. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66799-7_3
9. Han, Y., Kim, H., Ota, M., Seki, S.: Nondeterministic seedless oritatami systems and hardness of testing their equivalence. Nat. Comput. **17**(1), 67–79 (2018)
10. Han, Y.-S., Kim, H., Rogers, T.A., Seki, S.: Self-attraction removal from oritatami systems. In: Pighizzini, G., Câmpeanu, C. (eds.) DCFS 2017. LNCS, vol. 10316, pp. 164–176. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60252-3_13
11. Harel, D., Sardas, M.: An algorithm for straight-line drawing of planar graphs. Algorithmica **20**(2), 119–135 (1998)
12. Hofacker, I.L., Fontana, W., Stadler, P.F., Bonhoeffer, L.S., Tacker, M., Schuster, P.: Fast folding and comparison of rna secondary structures. Monatshefte für Chemie / Chemical Monthly **125**(2), 167–188 (1994)
13. Ota, M., Seki, S.: Rule set design problems for oritatami system. Theor. Comput. Sci. **671**, 16–35 (2017)
14. Xayaphoummine, A., Viasnoff, V., Harlepp, S., Isambert, H.: Encoding folding paths of RNA switches. Nucleic Acids Res. **35**(2), 614–622 (2007)