Bit string bifurcation by cotranscriptional folding*

Yusei Masuda, Shinnosuke Seki**, and Yuki Ubukata

Department of Computer and Network Engineering, The University of Electro-Communications, 1-5-1, Chofugaoka, Chofu, Tokyo, 1828585, Japan.

Abstract. We demonstrate cotranscriptional folding of a finite part of the Heighway dragon, a fractal also-known as the paperfolding sequence $P = RRLRRLLR\cdots$ by an oritatami system. The i-th element of P can be obtained by feeding i in binary to a 4-state deterministic finite automaton with output (DFAO). We implement this DFAO and a bit-string bifurcator as modules of oritatami system. Combining them with a known binary counter module yields the target oritatami system.

Keywords: Heighway dragon, Fractal, Cotranscriptional folding, Oritatami system, Automatic sequence, Bitstring bifurcation

1 Introduction

Molecular shape self-assembly is an engineering technology that provides a design of molecules that organize into a complex target shape due to some simple foundational phenomenon in which molecules interact with each other locally without or with little external control. The foundational phenomenon studied most intensively so far is the coalescence of DNA rectangular tiles via their sticky ends (see, e.g., [5, 10, 11]). Geary, Rothemund, and Andersen have recently shed light on self-assembly capability of another phenomenon called *RNA cotranscriptional folding* [8]. An RNA sequence is synthesized (*transcribed*) sequentially out of its template (complementary) DNA sequence by an RNA polymerase enzyme. The sequence thus synthesized is called a transcript (for details, see, e.g., [2]). While being transcribed, the transcript folds into an intricate tertiary structure. This is the cotranscriptional folding. In [8], the authors demonstrated an architecture of DNA template sequence whose RNA transcript thus folds into an RNA rectangular tile as intended in a laboratory. *Oritatami system* is a novel mathematical model of RNA cotranscriptional folding [7]. Computational aspects of RNA cotranscriptional folding such as counting in binary and Turing universality have been studied [4, 6, 7, 9], but oritatami systems have not been used to self-assemble a shape.

* This work is in part supported by JST Program to Disseminate Tenure Tracking System, MEXT, Japan, No. 6F36 and by JSPS Grant-in-Aid for Young Scientists (A) No. 16H05854 to S. S.

** Corresponding author s.seki@uec.ac.jp

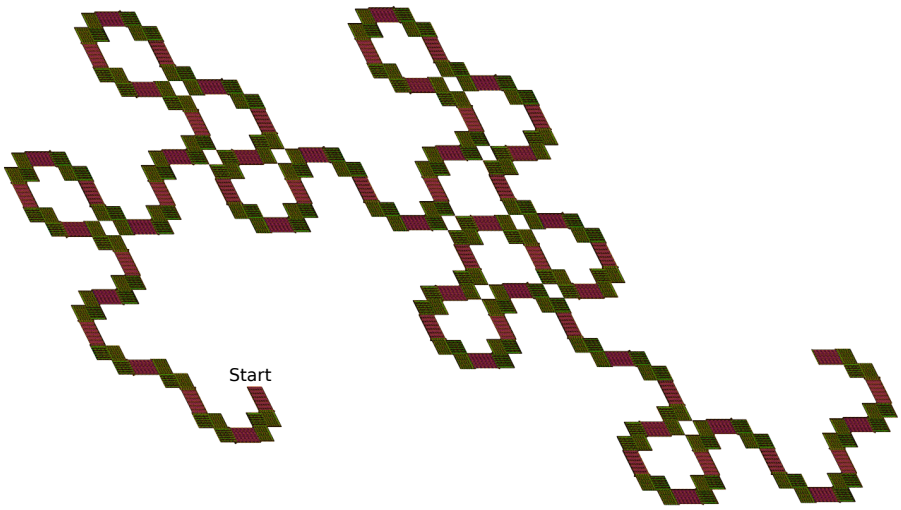

Fig. 1. Contranscriptional folding of 6-bit Heighway dragon by the proposed oritatami system.

This paper proposes an oritatami system that self-assembles an n-bit fraction of the Heighway dragon (see Fig. 1 for 6-bit fraction), a kind of self-similar fractal dragon curve, for an arbitrary integer $n \geq 1$. The n-bit fraction consists of the first 2^n turns of the Heighway dragon. This fractal is also called the *paperfolding sequence* named after its generating method: fold a paper slip in the middle, fold the result further in the middle, and so on. The result being unfolded consists of left and right turns as $P = RRLRRLLR\cdots$. This is the paperfolding sequence. There exists a deterministic finite automaton with output (DFAO) that returns the i-th element of P when the binary representation of i is fed into it from the least significant bit (LSB) [3]. We implement a binary counter, this DFAO, and a turning mechanism in oritatami systems that share a common interface through which they can interlock with each other and propagate a current count i. The turning mechanism is implemented by combining three instances of a module that bifurcates a given bit string into two directions and guides the system along one of them according to the output of the preceding DFAO. Since the oritatami binary counter is an existing technology [7], we will explain the DFAO and turning mechanism in this paper.

A java-script program to execute this oritatami system and websites on which this program is executable are freely available [1].

2 Preliminaries

Let Σ be a set of types of abstract molecules, or *beads*, and Σ^* be the set of finite strings of beads. Let $w = a_1a_2\cdots a_n$ be a string of length n for some

integer n and bead types $a_1, ..., a_n \in \Sigma$. The *length* of w is n, denoted by $|w|$. For two indices i, j with $1 \leq i \leq j \leq n$, we let $w[i, j]$ refer to the substring $a_i a_{i+1} \cdots a_{j-1} a_j$. if $i = j$, then we express $w[i, i]$ as $w[i]$.

Let $\mathcal{H} \subseteq \Sigma \times \Sigma$ be a symmetric relation, specifying between which types of beads can form a hydrogen-bond-based interaction (h-interaction for short). This is called the *ruleset*.

Oritatami systems operate on the triangular lattice. The *grid graph* (V, E) of the lattice is the graph whose vertices correspond to the lattice points and connected if the corresponding lattice points are at unit distance. A *path* in the grid graph is a sequence $P = p_1 p_2 \cdots p_n$ of *pairwise-distinct* points $p_1, p_2, \cdots, p_n \in V$ such that $p_i, p_{i+1} \in E$ for all $1 \leq i < n$.

A *conformation* C is a triple of a *path* $P = p_1 p_2 \cdots p_n$, a *word* w of length n, and a set H of h-interactions, where $H \subseteq \{\{i, j\} \mid 1 \leq i, i+2 \leq j\}$ and $\{i, j\} \in H$ implies that the i-th and j-th beads of the path, i.e., p_i and p_j, form an h-interaction between them. The condition $i + 2 \leq j$ represents the topological restriction that two beads next to each other along the path cannot form an h-interaction between them. For an integer $\alpha \geq 1$, let $\mathcal{C}_{\leq \alpha}$ be the set of all conformations whose arity is at most α. A conformation C is *valid with respect to* $\mathcal{H}$, or $\mathcal{H}$-valid in short, if for all $\{i, j\} \in H, (w[i], w[j]) \in \mathcal{H}$. In a context with one fixed ruleset, only valid conformations w.r.t. the ruleset are considered.

Given a ruleset $\mathcal{H}$ and an $\mathcal{H}$-valid finite conformation $C_1 = (P, w, H)$, we say that another conformation C_2 is an *elongation of* C_1 *by a bead* $a \in \Sigma$ if $C_2 = (P \cdot p, w \cdot a, H \cup H')$ for some lattice point p not along the path P and (possibly empty) set of h-interactions $H' \subseteq \big\{\{i, |w|+1\} \mid 1 \leq i \leq |w|, (w[i], a) \in \mathcal{H}\big\}$. Note that C_2 is also $\mathcal{H}$-valid. This operation is recursively extended to the elongation by finite sequence of beads as: for any conformation C, $C \xrightarrow{\mathcal{H}}_{\lambda} C$; and for a finite sequence of beads $w \in \Sigma^*$ and a bead $b \in \Sigma$, a conformation C_1 is elongated to a conformation C_2 by the sequence wb, written as $C_1 \xrightarrow{\mathcal{H}}_{wb} C_2$, if there is a conformation C' that satisfies $C_1 \xrightarrow{\mathcal{H}}_{w} C'$ and $C' \xrightarrow{\mathcal{H}}_{b} C_2$.

2.1 Oritatami system

An *oritatami system* is a 5-tuple $\Xi = (\mathcal{H}, \alpha, \delta, \sigma, w)$, where $\mathcal{H}$ is a *ruleset*, α is an *arity*, $\delta \geq 1$ is a parameter called the *delay*, σ is an initial valid conformation of arity α called the *seed*, upon which its *transcript* $w \in \Sigma^* \cup \Sigma^w$ is to be folded by stabilizing beads of w one at a time so as to minimize energy collaboratively with the succeeding $\delta - 1$ nascent beads. The energy of a conformation $C = (P, w, H)$, denoted by $\Delta G(C)$, is defined to be $-|H|$; the more h-interactions a conformation has, the more stable it gets. The set $\mathcal{F}(\Xi)$ of conformations *foldable* by this system is recursively defined as: the seed σ is in $\mathcal{F}(\Xi)$; and provided that an elongation C_i of σ by the prefix $w[1...i]$ be foldable (i.e., $C_0 = \sigma$), its further elongation C_{i+1} by the next bead $w[i+1]$ is foldable if

$$C_{i+1} \in \operatorname*{arg\,min}_{\substack{C \in \mathcal{C}_{\leq \alpha} s.t. \\ C_i \xrightarrow{\mathcal{H}}_{w[i+1]} C}} \min\Big\{\Delta G(C') \mid C \xrightarrow{\mathcal{H}^*}_{w[i+2...i+k]} C', k \leq \delta, C' \in \mathcal{C}_{\leq \alpha}\Big\}. \quad (1)$$

We say that the bead $w[i+1]$ and the h-interactions it forms are *stabilized* according to C_{i+1}. Note that an arity-α oritatami system cannot fold any conformation of arity larger than α. A conformation foldable by Ξ is *terminal* if none of its elongations is foldable by Ξ.

The oritatami system Ξ is *deterministic* if for all $i \geq 0$, there exists at most one C_{i+1} that satisfies (1). Thus, a deterministic oritatami system folds into a unique terminal conformation.

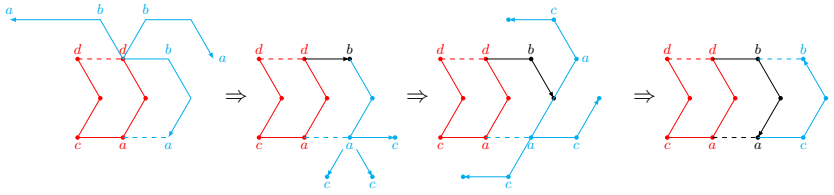

Fig. 2. Progression of a glider by distance 1.

Example 1. Let us explain the behavior of oritatami system using a motif called the *glider*. Consider a delay-3 oritatami system whose seed is colored in red in Fig. 2 (Left), primary structure is $b \bullet ac \bullet bd \bullet c \cdots$, and the ruleset is $\mathcal{H} = \{(a,a),(b,b),(c,c),(d,d)\}$.

By the fragment $b \bullet a$, the seed can be elongated in many ways; three of them are shown in Fig. 2 (left). Only the a can form a new h-interaction, and for that, the bead must be located to the east of the last bead of the seed; thus, the b is stabilized there, as shown in Fig. 2. Then the next bead c is transcribed. The sole other c around is too far for the c to get adjacent to. Thus, the only way for the fragment $\bullet ac$ to form an h-interaction is to put the two a's next to each other as before, and for that, the $\bullet$ must be located to the southeast of the preceding b. The next bead to be transcribed is inert, and hence, the a is to be adjacent to the a on the seed. The first three beads $b, \bullet, a$ have been thus stabilized as shown in Fig. 2 (right), and the glider has moved forward by distance 1. It is easily induced inductively that gliders of arbitrary "flight distance" can be folded.

Gliders also provide a medium to propagate 1-bit information at arbitrary distance. The height (top or bottom) of the first bead determines whether the last bead is stabilized top or bottom after flying a given distance. For instance, the glider in Fig. 2 launches top and thus its last bead (the second b) also comes top after traveling the distance 2. The oritatami system we will propose exploits this information-carrying capability.

2.2 Heighway dragon

The Heighway dragon is one of fractals. It can be described by a binary sequence that is called "paperfolding sequence" [3].

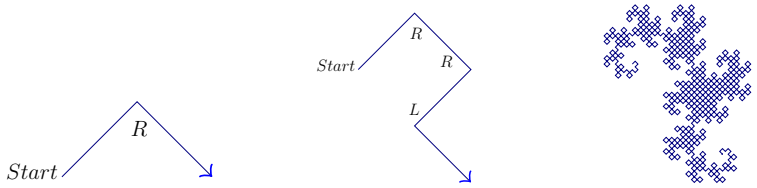

Fig. 3. Heighway dragon: (left) One Fold, (center) Two Folds, (right) Ten Folds [3].

The paperfolding sequence is named after the following operation to generate it. First, take a piece of paper and fold it in half lengthwise, then fold the result in half again, and so on. Next, unfold the paper. The resulting sequence $(P_i)_{i\geq 0}$ of left turns (L) and right turns (R) is a paperfolding sequence. For example, after one fold, we get the pattern in Fig. 3 (left). Two folds and ten folds result in the respective patterns in Fig. 3 (center) and (right).

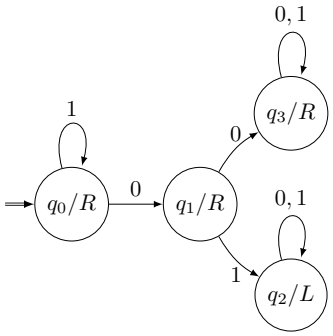

Fig. 4. DFAO for Paperfolding sequence [3]

The paperfolding sequence can be generated by the deterministic finite automaton with output (DFAO) in Fig. 4 (automatic sequences). Each state of the DFAO is assigned with L or R as an output. For $i \geq 0$, n-th the element of the sequence, i.e, P_i, can be computed by feeding the DAFO with the base-2 representation of i from its LSB as the letter assigned to the last state reached. Here are the first few terms of the sequence thus obtained.

$$\begin{aligned} i &= 0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\cdots \\ P_i &= R\ R\ L\ R\ R\ L\ L\ R\ R\cdots \end{aligned}$$

This sequence can be interpreted as the Heighway dragon as shown in Fig. 3.

3 Main results

We propose a deterministic oritatami system that $\Xi = (\mathcal{H}, \alpha, \delta, \sigma, w)$ folds into the n-bit Heighway dragon. The dragon it folds into is actually slanted as illustrated in Fig. 1, which is more natural than the conventional (upright) one to be folded over the triangular grid. The oritatami system employs about 2000 bead types, not depending on n. Its delay δ is set to 3 and its arity α is also set to 3.

This oritatami system is periodic, that is, its transcript w is a periodic sequence. Recall that we can draw the Heighway dragon by repeating the following processes:

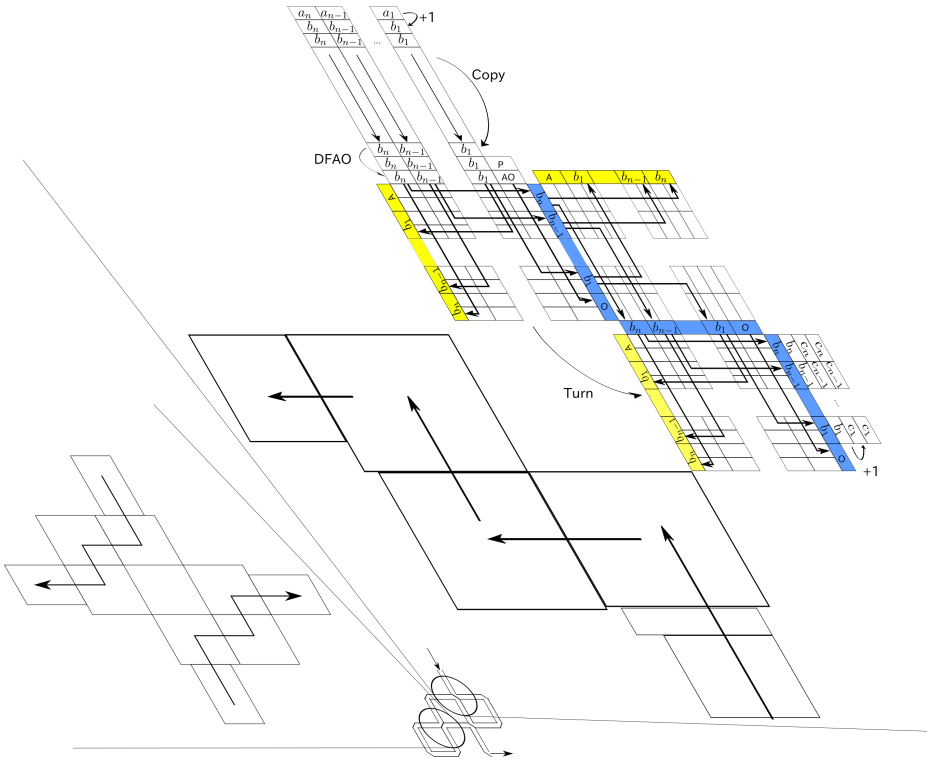

Fig. 5. Folding of one segment plus turn of the Heighway dragon, flow of information through it, and two ways of collision avoidance between two turns.

1. $i \leftarrow i + 1$ (increment)
2. copy i (drawing a segment)
3. compute P_i
4. turn according to P_i

One period of the system consists of four subsequences: Counter, Xerox, DFAO, and Turner, that handle these four processes, respectively. These subsequences fold as abstracted in Fig. 5 into one segment plus one turn of the Heighway dragon. The length of one period is proportional to the square of n. The initial value 0 of i is encoded in the seed of the system, which replaces the first Counter.

Counter and Xerox are implemented as done in the binary counter of [7].[1] They increment/copy i represented in binary. The resulting i is encoded at the bottom of them as a sequence of bead types; different sequences are thus exposed for 0 and 1.

In order to explain how DFAO and Turner are implemented, we have to raise two issues of significance specific to the folding of Heighway dragon by oritatami

[1] The oritatami system of [7] uses a different but more tractable dynamics than the dynamics used in this paper.

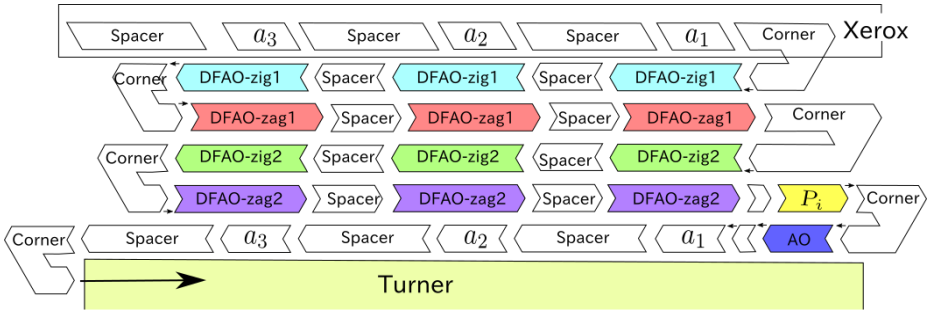

Fig. 6. The module-level abstraction of the folding of DFAO.

Fig. 7. (left) The possible two conformations of PFS: PFS-L, PFS-R. (right) The possible two conformations of AO.

system. The first issue arises from that the dragon is slanted. The slanted dragon involves two types of left turn as well as two types of right turn: acute and obtuse. Does this mean that we have to implement Turner that is capable of all the four types of turn? That is not the case fortunately. Observe that after the (slanted) vertical segment, certainly the left turn is obtuse and the right turn is acute, while after the horizontal segment the left turn is acute and the right turn is obtuse. In addition, we know *a priori* which segments are vertical and which are horizontal; indeed, vertical segments and horizontal segments occur alternately on the Heighway dragon. We can hardcode a proper interpreter of the output of DFAO, that is, P_i, as acute or obtuse in the oritatami system. This requires two types of such interpreter, one of which converts L into acute and R into obtuse, and the other of which does the other way around. These interpreters make it enough for us to provide Turner with just two functions: turning acutely or turning obtusely.

The second issue happens when the Heighway dragon makes a turn where it has already made another turn, that is, these two turns share a point. By definition, oritatami systems cannot put a bead at a point occupied by another bead. In oritatami systems, these turns need be mimicked so as for them not to share any point. As shown in Figs. 1 and 5, the proposed system implements an acute turn by having three Turners turn acutely one after another, and an obtuse one by having three Turners turn rather obtusely.

3.1 DFAO

DFAO receives the current count i from Xerox, computes P_i, interprets it properly either as A or O, and outputs it together with the count i.

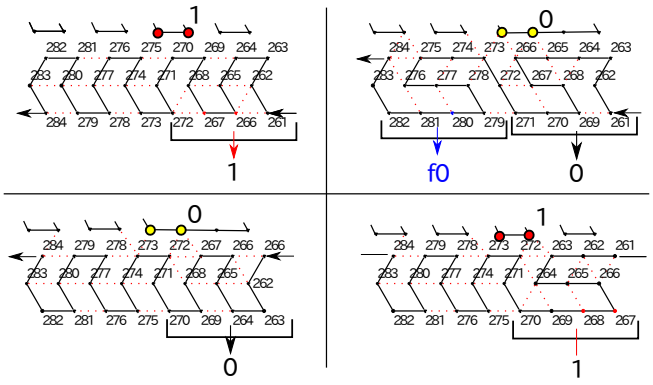

Fig. 8. The possible four conformations of DFAO-zig1: (top) Dzig1-1, Dzig1-f0, (bottom) Dzig1-20, and Dzig1-21.

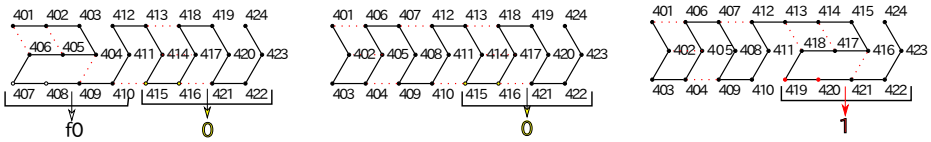

Fig. 9. The possible three conformations of DFAO-zag1: Dzag1-f0, Dzag1-0, and Dzag1-1 from left.

DFAO is composed of the six modules: "DFAO-zig1", "DFAO-zag1", "DFAO-zig2", "DFAO-zag2", "PFS", "AO" (or "$\overline{\text{AO}}$"). It folds as abstracted in Fig. 6. What the system does in the first zig-zag is to have DFAO-zig1s and DFAO-zag1s read the current count i from its LSB and "mark" the first 0, while propagating i. In the second zig-zag, it employs DFAO-zig2s and DFAO-zag2s to check whether the automaton transitions to the state q_2 ($P_i = L$) or else ($P_i = R$). The second zag is to end at the top if $P_i = L$ or at the bottom if $P_i = R$. PFS takes one of the two conformations in Fig. 7 and outputs P_i downward. In vertical segments, PFS is followed by AO, while in horizontal segments, it is followed by $\overline{\text{AO}}$. AO interprets the PFS' output $P_i = R$ as acute and $P_i = L$ as obtuse, as shown in Fig. 7. $\overline{\text{AO}}$ interprets them other way around. Let us explain briefly how the modules in the zigzags fold to fulfill their roles.

DFAO-zig1s detect the first 0 in two phases. Phase1 is to copy all the 1's before the first 0 and Phase2 is to copy 0s and 1's after the first 0. These phases are distinguished by the first bead of a module (in Phase1 it is at the bottom, while in Phase2 it is at the top, as suggested in Fig. 8). That is, in Phase1, DFAO-zig1s take the conformation Dzig1-1 (the top left conformation in Fig. 8). At the first 0, a DFAO-zig1 takes Dzig1-f0, starting at the bottom but ending at the top (phase transition). Each of the succeeding DFAO-zig1s takes one of the remaining two conformations Dzig1-20 and Dzig1-21 for copy in Phase2. Note that there is a cushion between two DFAO-zig1s called *spacer*. Spacers have been

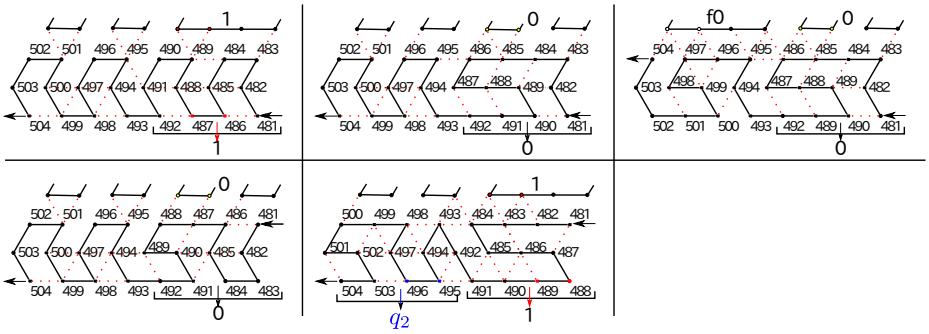

Fig. 10. The possible five conformations of DFAO-zig2: (top) Dzig2-1, Dzig2-0, Dzig2-f0, (bottom) Dzig2-f00, and Dzig2-f01.

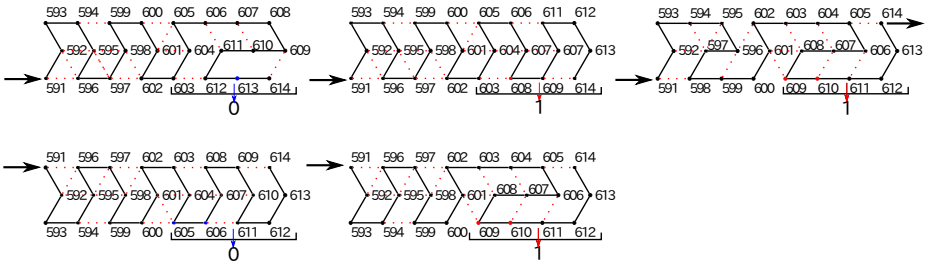

Fig. 11. The possible five conformations of DFAO-zag2: (top) Dzag2-L0, Dzag2-L1, Dzag2-T1, (bottom) Dzag2-R0, and Dzag2-R1.

already used to prevent modules of same kind from interfering undesirably. They are implemented as a glider (see Example 1), hence capable of propagating 1bit on which phase the system is in. In the first zag, DFAO-zag1s propagate 0's, 1's, and the first 0 by taking the proper one of the three conformations in Fig. 9.

In the second zig, DFAO-zig2s check whether the first 0 is followed by 0 or 1, being read from LSB. DFAO-zig2s first copy all the 1's before the first 0 by taking the conformation Dzig2-1 (top left in Fig. 10). The next letter is the first 0, which is distinguished from other 0's by the marker f0. When this module starts to fold at the bottom and reads 0, it can take two conformations Dzig2-0 and Dzig2-f0. These conformations share the first half. The marker f0 has the second half fold so as to end at the top as in Dzig2-f0. The next DFAO-zig2 therefore starts to fold at the top so that it takes one of the two conformations Dzig2-f00 and Dzig2-f01 depending on the input. Recall that reading 1 here is equivalent to transitioning to q_2, that is, $P_i = L$. These conformations end at the bottom. The remaining 0's and 1's are copied by Dzig2-0 and Dzig2-1, respectively. The second zag starts at the bottom and copy 0's and 1's by the two conformations Dzag2-L0 and Dzag2-L1 of DFAO-zag2 (top left and center in Fig. 11) until a DFAO-zag2 encounters a 1 marked q_2, if any. Such DFAO-zag2 takes the special conformation Dzag2-T1 and ends at the top, letting the remaining DFAO-zag2s

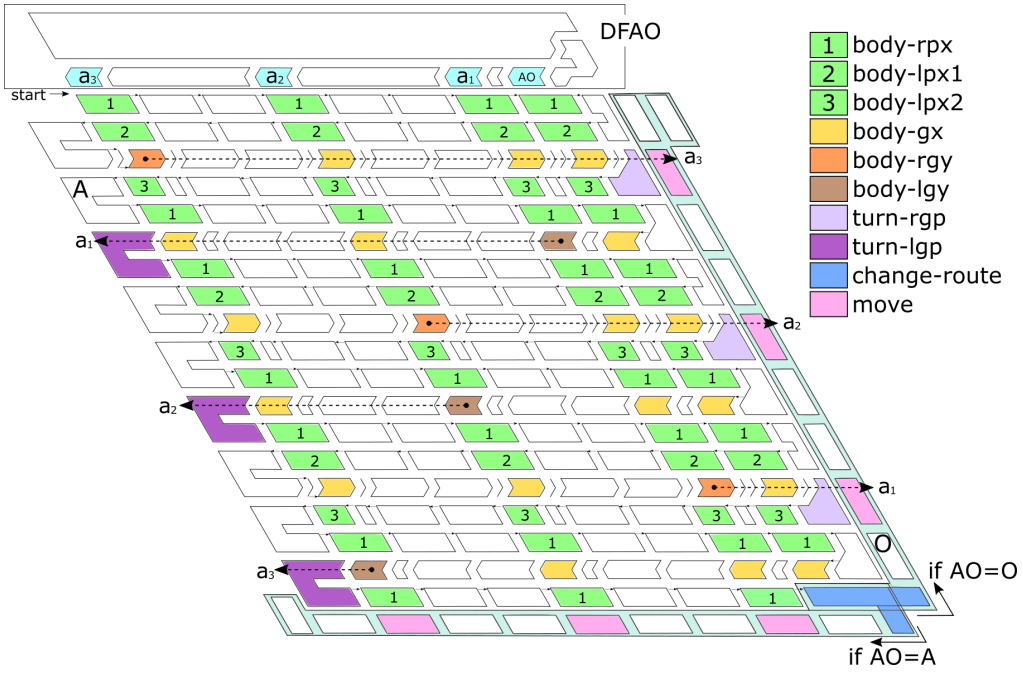

Fig. 12. The module-level abstraction of folding of Turner. All the white modules in the middle are spacers, some of which are implemented in the shape of parallelogram instead of glider.

rather take Dzag2-R0 and Dzag2-R1 for copying, which end at the top. As such, the second zag can feed P_i to PFS as explained before, while propagating the current count i.

3.2 Turner

Turner consists of two parts: bit string bifurcator and steering arm (colored in blue in Fig. 12).

The bifurcator sends binary information as in Fig. 5 while folding into zigzags. For that, it employs four types of module that propagates 1bit vertically (body-rpx, body-lpx1, and body-lpx2 in Fig. 12), that lets 1bit cross another 1bit (body-gx), that forks 1bit vertically and horizontally (body-rgy, body-lgy), and that undergoes transition from a zig to a zag or from a zag to a zig and exposes 1bit outside (turn-rgp, turn-lgp). The first two of them have already been implemented (see, e.g., [9]) so that we shall explain the others.

The module body-rgy takes one of the two conformations in Fig. 13 (left) depending on the 1bit encoded in the two beads above. Output below, the 1bit is encoded as a type of the second bead from left, while output right, it is encoded as whether this module ends top or bottom. Its zag-variant, body-lgy, is implemented analogously; for its conformations, see Fig. 13 (right).

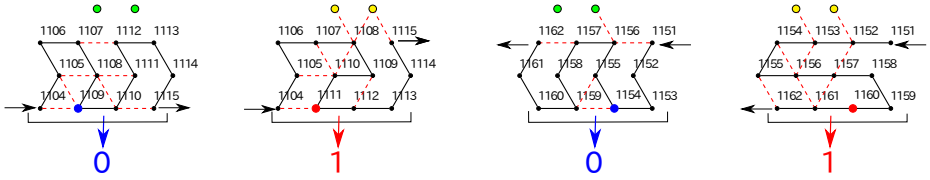

Fig. 13. (left) The possible two conformations of body-rgy and (right) of body-lgy.

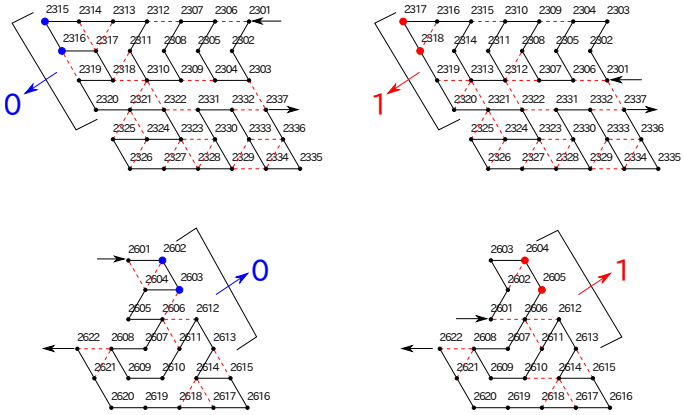

Fig. 14. (top) The possible two conformations of turn-lgp and (bottom) of turn-rgp.

The 1bit forked rightward by a body-rgy transfers till the end of the zig without being jammed because all remaining modules in the zig are designed in such a way that they start and end at the same height like the glider. The module turn-rgp receives the 1bit (top or bottom), and exposes it by taking one of the two comformations in Fig. 14 (bottom). The module turn-lgp functions analogously in zags as being holded in Fig. 14 (top).

The bifurcator also propagates a signal, A or O, fed by the DFAO, to tell the steering arm which way it should take. Specifically, the signal has the first module, change-route, of the steering arm take one of the two conformations in Fig. 15, guiding the rest of the steering arm in the ordered direction. The steering arm is provided with move modules, which let the bifurcated bit string pass through. Note that the Turner does not have to bifurcate AO. Indeed, the second Turner is supposed to turn in the same manner as the first one. It hence suffices to append A and O to the bifurcated bit strings on the acute side and obtuse side, respectively, as shown in Fig. 12.

Remark 1. In fact, as suggested in Fig. 12, the bifurcator outputs the bit string also downward. That is, it bifurcates a given bit string into three directions. This provides a more space-efficient way to replicate a bit string many-folds.

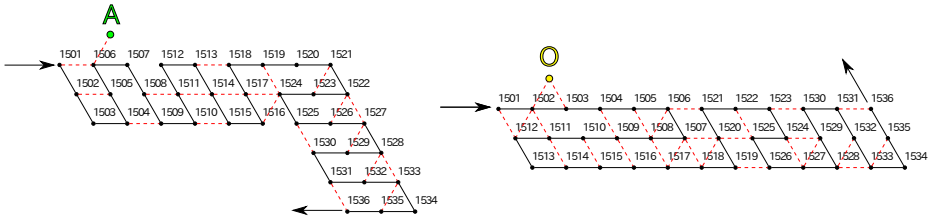

Fig. 15. The possible two conformations of change-route.

Acknowledgements

We would like to thank Hwee Kim and Aleck Johnsen for their helpful advices and discussions.

References

1. `https://wolves13.github.io/Heighway_dragon.html,https://wolves13.github.io/one_segment/One_segment_simu.html,https://github.com/wolves13/wolves13.github.io`
2. Alberts, B., Johnson, A., Lewis, J., Morgan, D., Raff, M., Roberts, K., Walter, P.: Molecular Biology of the Cell. Garland Science, 6 edn. (2014)
3. Allouche, J.P., Shallit, J.: Automatic Sequences: Theory, Applications, Generalizations. Cambridge University Press (2003)
4. Elonen, A.: Molecular folding and computation. Bachelor thesis, Aalto University (September 2016)
5. Evans, C.G.: Crystals That Count! Physical Principles and Experimental Investigations of DNA Tile Self-Assembly. Ph.D. thesis, California Institute of Technology (2014)
6. Geary, C., Meunier, P.E., Schabanel, N., Seki, S.: Oritatami cotranscriptional folding is Turing universal, in preparation
7. Geary, C., Meunier, P.E., Schabanel, N., Seki, S.: Programming biomolecules that fold greedily during transcription. In: Proc. MFCS 2016. LIPIcs, vol. 58, pp. 43:1–43:14 (2016)
8. Geary, C., Rothemund, P.W.K., Andersen, E.S.: A single-stranded architecture for cotranscriptional folding of RNA nanostructures. Science 345(6198), 799–804 (2014)
9. Han, Y.S., Kim, H., Ota, M., Seki, S.: Nondeterministic seedless oritatami systems and hardness of testing their equivalence. In: Proc. DNA 22. pp. 19–34. No. 9818 in LNCS (2016)
10. Patitz, M.J.: Self-assembly of fractals. In: Encyclopedia of Algorithms, pp. 1918–1922. Springer (2016)
11. Rothemund, P.W.K., Papadakis, N., Winfree, E.: Algorithmic self-aseembly of DNA sierpinski triangles. PLoS Biol. 2(12), e424 (2004)

Appendix

The possible conformations of all the module unexplained.

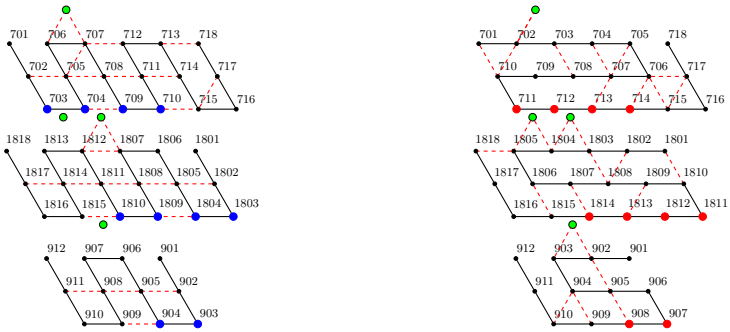

Fig. 16. (top) The possible two conformations of body-rpx, (middle) The possible two conformations of body-lpx1, (bottom) The possible two conformations of body-lpx2.

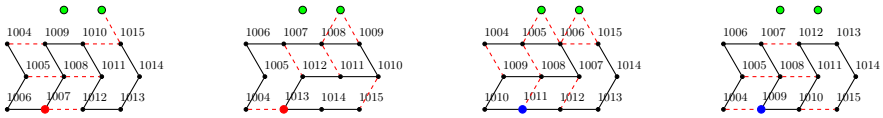

Fig. 17. The possible four conformations of body-gx.

Fig. 18. The possible two conformations of move.