Counting infinitely by Oritatami co-transcriptional folding

Kohei Maruyama[1⋆] and Shinnosuke Seki[1⋆⋆]

The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo, 1828585, Japan

Abstract. abstract

1 Introduction

Our body consists many proteins which is based on genetic sequence in DNA. DNA sequence has to be copied in order to make proteins because DNA is in cell nucleus. It is copied to RNA (and its sequence is called *transcript*) by a RNA polymerase and delivered to out of nucleus. A transcript consists four types nucleobases A, U, G, and C, and a nucleobase are called *bead* Moreover, they form hydrogen bonds mainly between A and U, and between G and C, then they begin to form at the same time synthesizing. Once the bead is fixed after that it can not change the position. By repeating this method, transcript form a 3D shape.

A shape of RNA depends on the DNA sequence. Geary, Rothemund, and Andrsen have succeeded in experimentally making 2D RNA tiles (Fig.1) by repeatedly copying a specific DNA sequence [3]. Then, Oritatami System was designed as a mathematical model that the process of RNA production [1] and Fig.2 is the 2D RNA tile which is represented on Oritatami System. Oritatami system treats transcript and nucleobase as paths and vertices which called *bead*, respectively, moreover, they are arranged on a triangular grid. The system takes transcript as an input and output an terminal conformation. Geary et al. implemented binary counter in this system [1]. This counter is folded in zigzags, and each row represents a counted number. Then, this counter is given a bit width as an input to the system before transfer begins, and encounters nondeterminism if it carries beyond that bit width. Since it is running on deterministic system, it halts when it becomes nondeterministic.

In this paper, based on the implementation method of this counter, we implement a counter which defined the behavior that extends the bit width when it overflows.

⋆ Primary corresponding author (k.maruyama@uec.ac.jp)
⋆⋆ Secondary corresponding author (s.seki@uec.ac.jp)

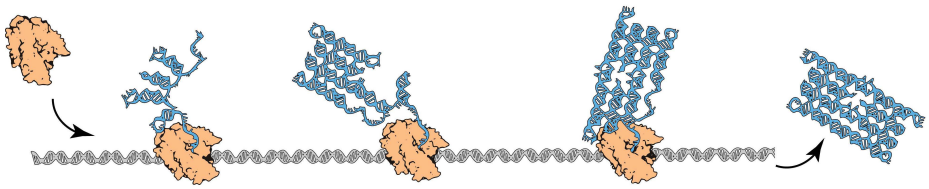

Fig. 1. A generated and stabilized 2D RNA tile from DNA

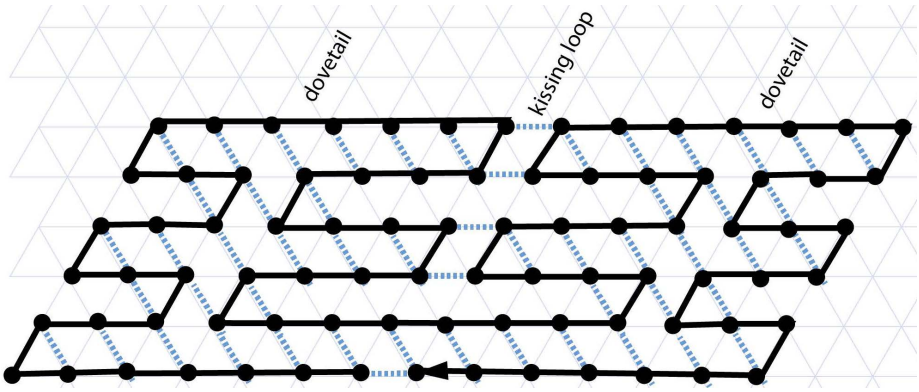

Fig. 2. The represented 2D RNA tile on Oritatami System

2 Preliminaries

Let Σ be a finite set of types of abstract molecules, or *beads*. A bead of type $a \in \Sigma$ is called an a-bead. By Σ^* and Σ^ω, we denote the set of finite sequences of beads and that of one-way infinite sequences of beads, respectively. The empty sequence is denoted by λ. Let $w = b_1 b_2 \cdots b_n \in \Sigma^*$ be a sequence of length n for some integer n and bead types $b_1, \ldots, b_n \in \Sigma$. The *length* of w is denoted by $|w|$, that is, $|w| = n$. For two indices i, j with $1 \leq i \leq j \leq n$, we let $w[i..j]$ refer to the subsequence $b_i b_{i+1} \cdots b_{j-1} b_j$; if $i = j$, then $w[i..i]$ is simplified as $w[i]$. For $k \geq 1$, $w[1..k]$ is called a *prefix* of w.

Oritatami systems fold their transcript, which is a sequence of beads, over the triangular grid graph $\mathbb{T} = (V, E)$ cotranscriptionally. We designate one point in V as the origin O of $\mathbb{T}$. For a point $p \in V$, let $\hexagon_p^d$ denote the set of points which lie in the regular hexagon of radius d centered at the point p. Note that $\hexagon_p^d$ consists of $3d(d+1)+1$ points. A directed path $P = p_1 p_2 \cdots p_n$ in $\mathbb{T}$ is a sequence of *pairwise-distinct* points $p_1, p_2, \ldots, p_n \in V$ such that $\{p_i, p_{i+1}\} \in E$ for all $1 \leq i < n$. Its i-th point is referred to as $P[i]$. Now we are ready to abstract RNA single-stranded structures in the name of conformation. A *conformation* C (over Σ) is a triple (P, w, H) of a directed path P in $\mathbb{T}$, $w \in \Sigma^*$ of the same length as P, and a set of h-interactions $H \subseteq \big\{\{i, j\} \bigm| 1 \leq i, i+2 \leq j, \{P[i], P[j]\} \in E\big\}$. This is to be interpreted as the sequence w being folded along the path P in

such a manner that its i-th bead $w[i]$ is placed at the i-th point $P[i]$ and the i-th and j-th beads are bound (by a hydrogen-bond-based interaction) if and only if $\{i,j\} \in H$. The condition $i+2 \leq j$ represents the topological restriction that two consecutive beads along the path cannot be bound. The *length* of C is defined to be the length of its transcript w (that is, equal to the length of the path P). A *rule set* $R \subseteq \Sigma \times \Sigma$ is a symmetric relation over Σ, that is, for all bead types $a, b \in \Sigma$, $(a,b) \in R$ implies $(b,a) \in R$. A bond $\{i,j\} \in H$ is *valid with respect to* R, or simply R-valid, if $(w[i], w[j]) \in R$. This conformation C is *R-valid* if all of its bonds are R-valid. For an integer $\alpha \geq 1$, C is *of arity* α if it contains a bead that forms α bonds but none of its beads forms more. By $\mathcal{C}_{\leq \alpha}(\Sigma)$, we denote the set of all conformations over Σ whose arity is at most α; its argument Σ is omitted whenever Σ is clear from the context.

The oritatami system grows conformations by an operation called elongation. Given a rule set R and an R-valid conformation $C_1 = (P, w, H)$, we say that another conformation C_2 is an elongation of C_1 by a bead $b \in \Sigma$, written as $C_1 \xrightarrow{R}_b C_2$, if $C_2 = (Pp, wb, H \cup H')$ for some point $p \in V$ not along the path P and set $H' \subseteq \big\{\{i, |w|+1\} \;\big|\; 1 \leq i < |w|, \{P[i], p\} \in E, (w[i], b) \in R\big\}$ of bonds formed by the b-bead; this set H' can be empty. Note that C_2 is also R-valid. This operation is recursively extended to the elongation by a finite sequence of beads as: for any conformation C, $C \xrightarrow{R}{}^*_\lambda C$; 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 wb, written as $C_1 \xrightarrow{R}{}^*_{wb} C_2$, if there is a conformation C' that satisfies $C_1 \xrightarrow{R}{}^*_w C'$ and $C' \xrightarrow{R}_b C_2$.

An *oritatami system* (OS) $\Xi = (\Sigma, R, \delta, \alpha, \sigma, w)$ is composed of

- a set Σ of bead types,
- a rule set $R \subseteq \Sigma \times \Sigma$,
- a positive integer δ called the *delay*,
- a positive integer α called the *arity*,
- an initial R-valid conformation $\sigma \in C_{\leq \alpha}(\Sigma)$ called the *seed*, whose first bead is assumed to be at the origin O without loss of generality,
- a (possibly infinite) *transcript* $w \in \Sigma^* \cup \Sigma^\omega$, which is to be folded upon the seed 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 bonds a conformation has, the more stable it gets. The set $\mathcal{F}(\Xi)$ of conformations *foldable* by the system Ξ 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

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

Then we say that the bead $w[i+1]$ and the bonds it forms are *stabilized* according to C_{i+1}. The easiest way to understand this stabilization process should be the video available at `https://www.dailymotion.com/video/x3cdj35`, in which the Turing universal oritatami system by Geary et al. [2], whose delay is 3, is running. 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.

3 Folding an infinite binary counter

3.1 General idea

Basically, an folding process of counter is the same way as Geary et al [1]. The construction proceeds in zig-zags. Each pass is 3-rows thick and folds each part of the bead into a glider or turn module. Each pass extends to straight while folding into glider and folds its last module folds into turn module and starts next pass. The zig pass, folding from right to left, counts up in the current value of the counter. The zag pass, folding from left to right, formats and copy to downward the value which the zig pass represents.

The module is semantically divided into 4 parts, called modules:

- Module F (beads 1–30): the Format module
- Module L (beads 31–66): the Left-Turn module
- Module H (bead 67–96): the Half-Adder module
- Module R (bead 97–132): the Right-Turn module

Encoding The current value of the counter is encoded in standard binary with the most significant bit to the left. Binary is encoded into a specific folding of the module F and H. In zig pass, folding of module H corresponds to module F in a zag pass: namely H0 for 0, and H1 for 1. In addition, start position of modules represents the carry: carry = 0 if module H starts to fold in the top row; carry = 1 if module H start to fold from the bottom row. On the other hand in zag pass, folding of module F corresponding to module H in a zig pass: namely F0 for 0, and F1 for 1.

In the zig pass (←) Module H read from the row above the value encoded into folding in the module F during the previous zag-phase, and fold into a shape H00, H10, H01, H11. Hxc is corresponding to the case where x is the bit read in the row above and c is the carry. In the zig pass module R, F, and L just propagate the carry value to the next.

When the zig pass reaches the left most part, module L collaborates with above turn module and it is folded turn module which reverses the folding direction for next zag pass except carry = 1. At this time, if module L has a carry, folding of module L is glider and it go straight in order to expand bit width after that next module L turns to zag pass while folding turn module.

In the zag pass (→) Module F read form the row above Hxc, and fold into a shape F0, F1. Fy is corresponding to the case where $y = (x + c) \bmod 2$. There are no carry propagation and the module L, H, R fold into Lbc, Hn, Rb in this pass.

When the zag pass reaches the right most part. module R collaborates with above turn module and it is folded turn module which reverses the folding direction for next zag pass.

3.2 Example of folding first zigzag

Let's run an infinite counter from 1 bit width. The seed is encoded with "0", and the starts from bottom row, that is, giving a carry for the least significant bit.

The first zig pass (←) In Fig, 3, the zig pass folds as follows.

- [First module F (Light Green)]
  Normally, module H is folded above module F, and the seed corresponding to this module H now represents None. Then, module F forms a shape which represents None and propagates carry equal to 1.
- [First module L (Light Blue)]
  Since the seed here does not have a turn signal, module L forms glider and propagates carry while extending straight.
- [First module H (Pink)]
  The seed is encoded 0 and module folding starts from the bottom row, that is, the carry equal to 1. Then, HA folds into H01, which represents 1 and is output of carry equal 0.
- [First module R (Yellow) and 2nd module F]
  Both module form glider which start from top row and end to top row, in order to propagate carry equal 0.

The first zag pass (→) In Fig. 3, it is folded until second module F, and module L is transferred as the next transcript. Fig. 4 shows how zag pass is folded. Now, there is a Turn Signal at the left end of the seed, so L's transcript binds to its signal and loses the shape of glider in order to turn. In addition, the turned module L forms a Turn Signal at its end.

The zag pass folds as follows.

- [module H and module R]
  Since the module F of last module in zig pass represents None, module H folds into Hn also representing None. The module R form glider for margin.
- [module F, L, and H]
  There are three types of brick, H01, H10 and He1, representing 1 in module H, and different signals are defined as 1 any. Then, module F folds into brick F1 no matter which module H is above, representing 1. The module L folds glider and the module H folds brick Hn.

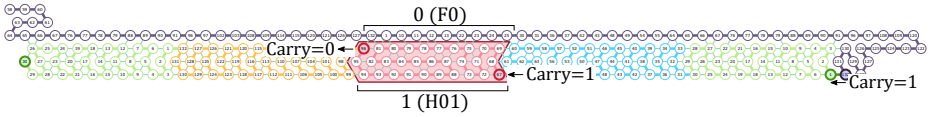

Fig. 3. The first zig pass. The seed is encoded with "0", and the counter increments it by giving a carry. Module H outputs "1" as a sum and cancels the carry.

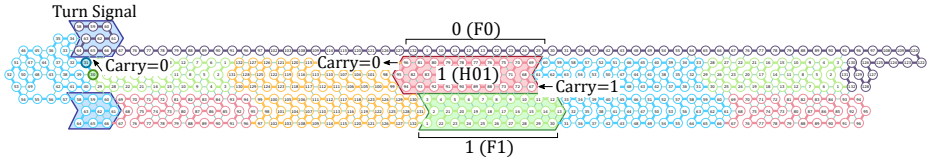

Fig. 4. Module L turns and start the first zag pass. Since there is a Turn Signal at the left end of the seed, when the carry is 0, module L turns and at the end of L forms Turn Signal. In zag pass, module F reads the output of module H and copies it down,

3.3 Conformations of Module H

The half adder module H forms five different paths as shown in Fig. 5. Module H reads the already folded bead (F0 or F1) at above as one of the inputs, and determines the other input depending on the folding starts (top or bottom). Module H forms four bricks of H00, H01, H10, and H11 for each input. Then, the bricks outputs the sum as beads of bottom row and also outputs carry depending on whether module H is folded top or bottom except He1. Brick He1 is the first folded module H after overflow. Since this counter uses three row of gliders as a self-standing method, module H consisting of 30 beads has an even width, and when it starts folding from the bottom, folding ends at the bottom. That is why brick He1 ends folding at the bottom and outputs carry equal to 0.

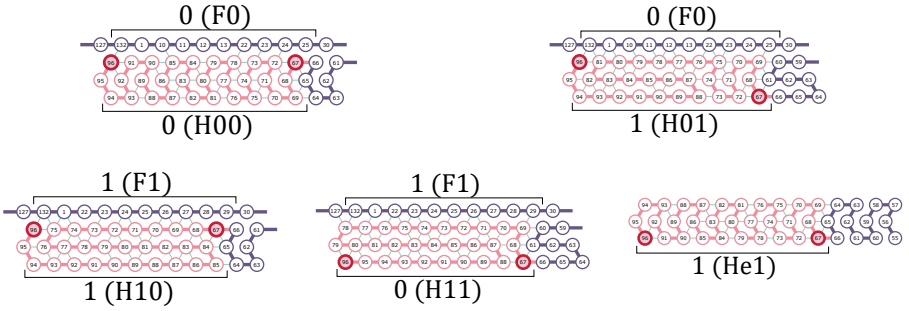

Fig. 5. The paths of module H in zig pass

3.4 Example of over flow

4 How to implement each module with oritatami system

Each module changes the output depending on the start position of folding and the environment.

References

1. 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)
2. Geary, C., Meunier, P.E., Schabanel, N., Seki, S.: Proving the Turing universality of oritatami cotranscriptional folding. In: Proc. ISAAC 2018. pp. 23:1 – 23:13 (2018)
3. 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)