Counting infinitely by oritatami co-transcriptional folding

Kohei Maruyama · Shinnosuke Seki

Abstract A fixed bit-width counter was proposed as a proof-of-concept demonstration of an oritatami model of cotranscriptional folding [Geary et al., Proc. MFCS 2016, LIPIcs 58, 43:1-43:14], and it was embedded into another oritatami system that self-assembles a finite portion of Heighway dragon fractal. In order to expand its applications, we endow this counter with capability to widen bit-width at every encounter with overflow.

1 Introduction

Counting is one of the most essential tasks for computing; as well known, the ability to count suffices to enable Turing universality [9]. Nature has been counting billions of days using molecular "circadian clockwork" which is "as complicated and as beautiful as the wonderful chronometers developed in the 18th century" [8]. Nowadays, developments in molecular self-assembly technology enable us to design molecules to count. Evans has demonstrated a DNA tile self-assembly system that counts accurately *in-vitro* in binary from

K. Maruyama
The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo, 1828585, Japan
E-mail: k.maruyama@uec.ac.jp

S. Seki
The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo, 1828585, Japan
École Normale Supériéure de Lyon, 46 allée d'Italie, 69007, Lyon, France
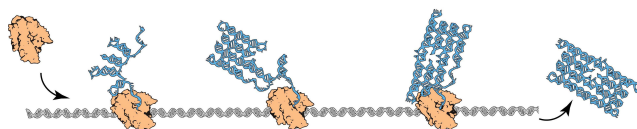E-mail: s.seki@uec.ac.jp

Fig. 1 RNA origami. RNA polymerase enzyme (orange) synthesizes the temporal copy (blue) of a gene (gray spiral) out of ribonucleotides of four types A, C, G, and U.

a programmed initial count until it overflows [3]. In its foundational theory of molecular self-assembly, such binary counters have been proved versatile, being used to assemble shapes of particular size [1,10], towards self-assembly of fractals [7], as an infinite scaffold to simulate all Turing machines in parallel in order to prove undecidability of nondeterminism in the abstract tile-assembly model [2], to name a few.

A fixed bit-width (finite) binary counter has been implemented as a proof-of-concept demonstration of the oritatami model of cotranscriptional folding [5]. As shown in Fig. 1, an RNA transcript folds upon itself while being transcribed (synthesized) from its corresponding DNA template strand. Geary et al. programmed a specific RNA rectangular tile structure into a DNA template in such a way that the corresponding RNA transcript *folds cotranscriptionally* into the programmed tile structure with high probability *in vitro* at room temperatures (*RNA origami*) [6]. An oritatami system folds a transcript of abstract molecules called *beads* of finitely many types over the 2-dimensional triangular lattice cotranscriptionally according to a rule set that specifies which types of molecules are allowed to bind once they are placed at the unit distance away. The transcript of the binary counter in [5] is of period 60 as ⓪−①−②−···−⑤⑧−⑤⑨−⓪−①··· and its period is semantically divided into two half-adder (HA) modules

$A = ⓪-①-\cdots-⑪$ and $C = ㉚-㉛-\cdots-㊶$ and two structural modules B and D, which are sandwiched by half-adder modules along the transcript. While being folded cotranscriptionally in zigzags, HA modules increment the current count i by 1, which is initialized on a linear *seed* structure, alike the Evans' counter, whereas structural modules B and D align HA modules properly and also make a turn at an end of the count i; B guides the transcript from a zig to a zag ($\hookrightarrow$) while D does from a zag to a zig ($\hookleftarrow$). This counter was embedded as a component of an oritatami system to self-assemble an arbitrary finite portion of Heighway dragon fractal [7]. Its applications are limited, however, by lack of mechanism to widen bit-width; its behavior is undefined when its count overflows. In this paper, we endow this counter, or more precisely, its structural module B, with the capability to widen the count by 1 bit at every encounter with overflow.

2 Preliminaries

Let Σ be a finite alphabet, whose elements should be regarded as types of abstract molecule, 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 $1 \le i \le j \le n$, 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 $k \ge 1$, $w[1..k]$ 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. For a point $p \in V$, let $\bigcirc_p^d$ denote the set of points which lie in the regular hexagon of radius d centered at the point p. Note that $\bigcirc_p^d$ 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 $1 \le i < n$. 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 $H \subseteq \big\{ \{i, j\} \mid 1 \le i, i+2 \le j, \{P[i], P[j]\} \in E \big\}$. 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 $i + 2 \le j$ 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 $\alpha \ge 1$, C is *of arity α* if it contains a bead that forms α bonds but none of its beads forms more. By $\mathcal{C}_{\le\alpha}(\Sigma)$, 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 $H' \subseteq \big\{ \{i, |w| + 1\} \mid 1 \le i < |w|, \{P[i], p\} \in E, (w[i], b) \in R \big\}$ 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) Ξ is a tuple $(\Sigma, R, \delta, \alpha, \sigma, w)$, where Σ and R are defined as above, while the other elements are a positive integer δ called *delay*, a positive integer α called *arity*, an initial R-valid conformation $\sigma \in C_{\le\alpha}(\Sigma)$ called the *seed*, and 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

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

$$(1)$$

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. [4], whose delay is 3, is running. This video is worth watching because a directed motif called a *glider*, which it features, is utilized for our infinite counter. 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.

Example 1 (Glider[1]) See Fig. 2 for a delay-3 oritatami system Ξ to fold a motif called a glider. Its transcript is a repetition of $a \bullet b b' \bullet a$ and its rule set R is $\{(a, a'), (b, b')\}$. Its seed is colored in red. The first 3 beads, $a \bullet b$, are transcribed and elongate the seed in all possible ways. The a-bead cannot bind or the second bead is inert according to R. The third bead, b, can bind to the b'-bead in the seed but for that, the a-bead must be located to the east of the previous a'-bead; it is thus stabilized there. Then the next bead, b', is transcribed. After the three steps, the third bead, b, is stabilized. It is not until then that its h-interaction with the b'-bead is also stabilized.

3 Folding an infinite binary counter

Between two consecutive overflows, the proposed system behaves in the same way as the finite binary counter proposed by Geary et al. [5]. Its transcript folds in a zigzag manner macroscopically (downward in figures throughout this paper). A zig, folding from right to left, increments the current value of the counter by 1. The succeeding zag, folding from left to right, formats the incremented value for the sake of next zig and copies it downward. Unlike the existing counter, when a zig encounters an overflow, it does not abort but rather extends the current value by 1 bit.

The transcript of our counter is periodic. Its period 1-2-3- $\cdots$ -132 is semantically divided into the following four subsequences, called *modules*: 1–30 (Format module or F; colored in green in figures), 31–66 (Left-Turn module or L; blue), 67–96 (Half-Adder module or H; red), and 97–132 (Right-Turn module or R; yellow). The transcript can be hence represented as $(FLHR)^*$ at the module level. Modules are to play their roles in expected environments by folding into respective conformations which should be pairwise-distinct enough to be distinguishable by other modules transcribed later. Such expected conformations are called a *brick*. For

module F encounters the four environments shown in Fig. 4 where it takes the four bricks `Fnt`, `Fnb`, `F0`, and `F1`, respectively. Here, by saying (an instance of) a module folds into (or takes) a brick in an environment, what we actually mean is that the rule set is designed so as for the transcript of the module to interact with itself as well as with the environment into that brick according to the oritatami dynamics (1). The whole system is designed to guarantee that each module is transcribed only in one of the environments it expects. This fact is illustrated in the *brick automaton*, which describes pairs of an environment and a brick as a vertex and transitions between them. Since this automaton is closed, it suffices to test whether for all pairs of an environment and a brick, the brick is folded in the environment. This test has been done *in-silico* using our simulator developed for this project. This brick automaton and all the certificates can be found at `https://komaruyama.github.io/oritatami-infinit-counter/`.

Seed and Encoding. The initial counter value is encoded as $b_{k-1} b_{k-2} \cdots b_1 b_0$ in binary on the seed in the following format

$$64-65-66-\left(\prod_{i=k-1}^{0} \left(w_{Hn} w_{Rb} w_{Fb_i} w_{Lbn} \right) \right) w_{Hn} \qquad (2)$$

where w_{Hn}, w_{Rb}, w_{F0}, w_{F1}, and w_{Lbn} are sequences of bead types exposed downward by modules H, R, F, L when they fold into bricks `Hn`, `Rb`, `Fb_i`, `Lbn`, respectively, which can be found in Figs. 6, 7, 4, and 5. For example, $w_{Hn} = 67-76-77-78-79-88-89-90-91-96$. The seed is examplified for $k = 1$ and $b_0 = 0$ in Fig. 8, where it is colord in purple.

Brick level overview Starting from the seed, this system cyclically transits four phases: zig ($\leftarrow$), left carriage-return ($\hookrightarrow$), zag ($\rightarrow$), and right carriage-return ($\leftarrow$). The prefix $(FLHR)^k F$ of the transcript folds into the first zig (recall that k is the bit-width of the initial value). In zigs in general, all the instances of modules F and H fold into bricks of width 10 and height 3, while all the instances of L and R fold into bricks of width 12 and height 3. Zigs thus turn out to be a linear structure of height 3. We can inductively observe that the i-th instance of H in the prefix is transcribed right below b_{i-1} encoded on the seed in the format (2) so that the H can "read" b_{i-1}. After the whole prefix thus has folded into the first zig, the next L is transcribed right below Turn Signal, which lets the L fold into a special brick for left carriage-return if the zig ended at the top (this occurs when $b_{k-1} b_{k-2} \cdots b_0 < 1^k$) (see Fig. 9). We should note that this special brick `Lcre` is provided with
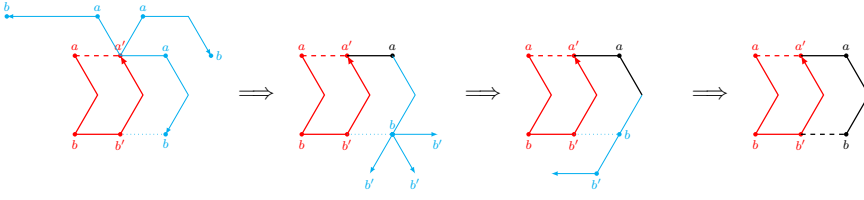
Fig. 2 Folding of a glider motif by a delay-3 deterministic oritatami system.

another Turn Signal for the sake of next left carriage-return. Having been thus carriage-returned, the succeeding subsequence $(HRFL)^k H$ of the transcript folds into the first zag. Even in zags instances of F and H fold into bricks of width 10 and height 3, while those of L and R fold into bricks of width 12 and height 3. As a result, zags turn out to be a linear structure of height 3. More importantly, instances of H and F are aligned thus vertically and alternately into columns (see Figs. 8-10), i-th of which from the right propagates the $(i-1)$-th bit of the counter value downward. After the whole subsequence has folded into the first zag, an instance of R is transcribed and folded into a special brick `Rcr` for right carriage-return due to the turn signal 125-124-123-122, which occurs also at the bottom of `Rcr` for the sake of next right carriage-return. This amounts to one cycle of the phase transition.

Increment of the counter In a zig, module H plays its primary role as a half-adder and carry transfers through instances of others (F, L, and R) from an instance of H to another for more significant bit. Carry transfers as a height for modules to start. In zigs, modules F, L, and R take the respective two bricks (`Fnt` and `Fnb` for F, `Lt` and `Lbn` for L, and `Rt` and `Rb` for R; see Figs. 4, 5, and 7), both of which start and end at the same height: one at the top while the other at the bottom. A zig is carried by being forced to start at the bottom by the last `Rcr` or the seed. Until the count overflows, module H encounters only four environments, which encode input 0 as w_{F0} or 1 as w_{F1} and carry or no-carry as of whether the module starts at the bottom or top, where it takes `H00`, `H01`, `H10`, and `H11`, respectively, as shown in Fig. 6 (`H`xc is folded when the input is x and the carry is given if $c = 1$ or not otherwise).

Let us see how the subsequence $(FLHR)^k F$ folds into a zig in order to count up; for $k = 1$ and the current value 0, see Fig. 8. The zig starts at the bottom, that is, being carried, and the carry transfers through the first instances of F and L in the way just explained toward the first instance of H. This H is thus fed with carry and folds into `H01` if the bit encoded above is 0, as illustrated in Fig. 8, or `H11` if the bit is rather 1. `H01` ends at the top, corresponding to canceling the carry out. This absence of carry transfers through the

succeeding modules leftward. As a result, the zig ends, or more precisely an instance of F ends folding at the bottom if the current value is overflowed (Fig. 10), or at the top otherwise (Fig. 8). An instance of L is to be transcribed next. It folds either into `Lcrn` for (normal) carriage-return unless the current value is overflowed, or into `Lbe` at an overflow.

Bit-width expansion at an overflow The counter of Geary et al. cannot handle a zig that ends at the bottom, that is, its behavior is undefined at an overflow. In contrast, module L in this infinite counter is designed so as to fold into `Lbe` in this situation in order to continue counting up (Fig. 11). Observe that the dent on Turn Signal made of 58, 63, and 64 is too far for module L or more precisely its beads 33 and 34 to interact with strongly enough to fold into `Lcrn`. `Lbe` is a self-sustaining conformation (glider) so that it can fold even if nothing is around, which occurs at that very moment. For the same reason, the following instances of H, R, and F fold into self-sustaining conformations `He1`, `Rb`, and `Fnb`, respectively (Figs. 11 and 12). Note that `He1` is essentially the same as `H00` but exposes the opposite side downward, which will be interpreted as the leading bit 1 after expansion in the next zag. When the next instance of L is transcribed, there is nothing around. Nevertheless, it does not fold into `Lbe` but folds into `Lcre` for carriage-return; how? It is guided by interaction between the beads 35, 36 along the transcript of L and the Turn Signal 28-27-22 above `Fnb` (Figs. 5 and 12). This signal is usually hidden geometrically by the previous zag, and hence, does no harm.

Formatting The value has been successfully incremented but it is not in the format (2) yet. In the upcoming zag, instances of F play their primary role to format 1 bit output by module H (recall that instances of H and F are aligned vertically and alternately). Both of the bricks of L for carriage-return, i.e., `Lcrn` and `Lcre`, end at the bottom so that zags start at the bottom. All modules start and end at the bottom in zags; note that nothing has to be transferred between modules. That is, instances of H, L, and R fold into `Hn`, `Lbn`, and `Rb`, respectively. Below the brick `H`xc, an instance of F folds into `F`y, where $y = (x + c) \mod 2$.
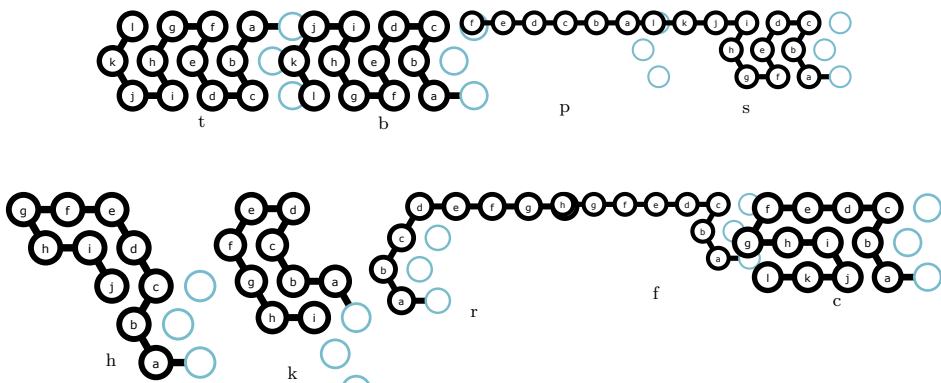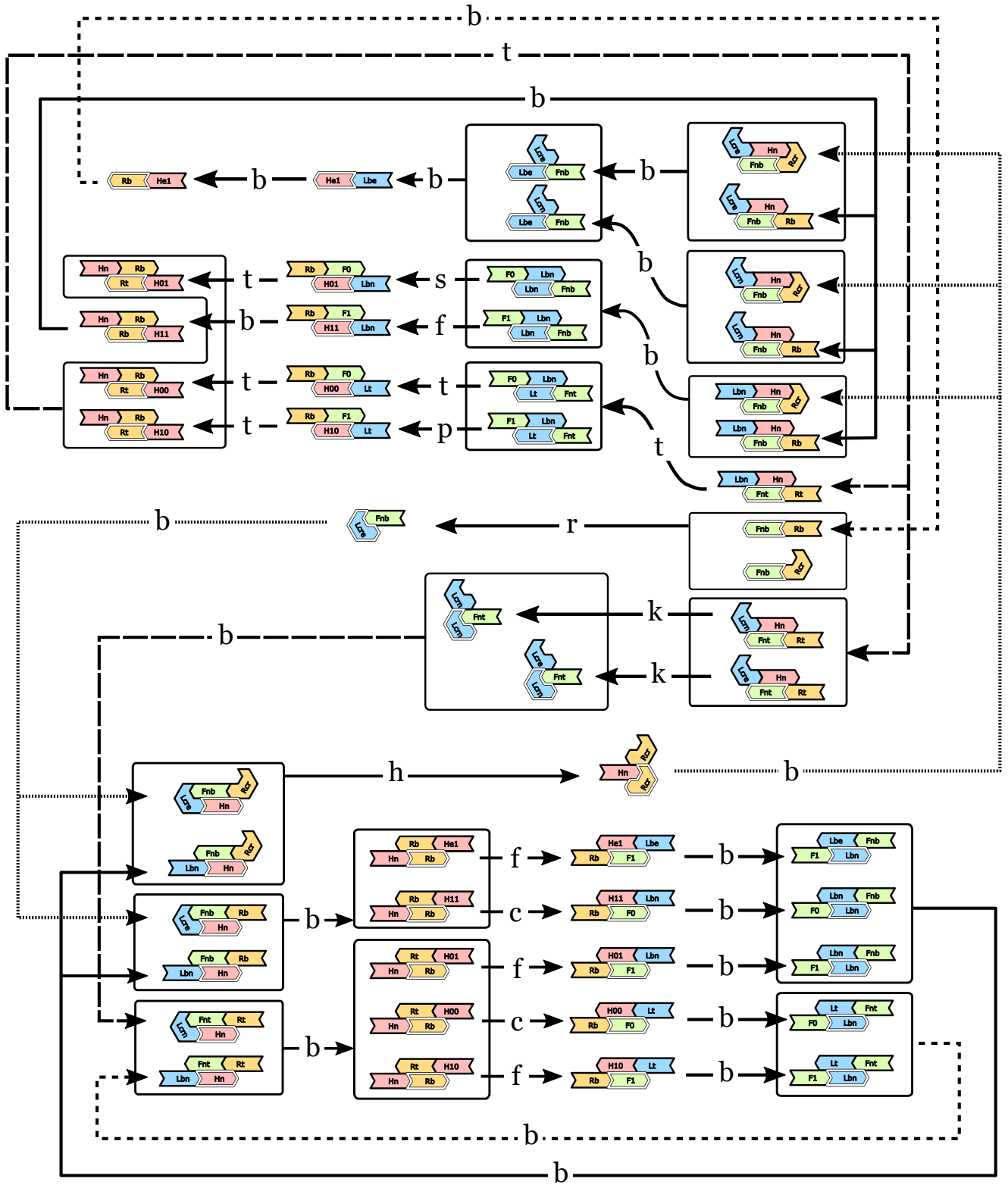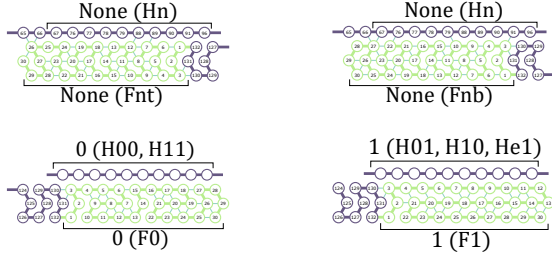
Fig. 4 All the four bricks of module F: The two bricks at the top, Fnt and Fnb, are for zigs while the others, F0 and F1, are for zags.
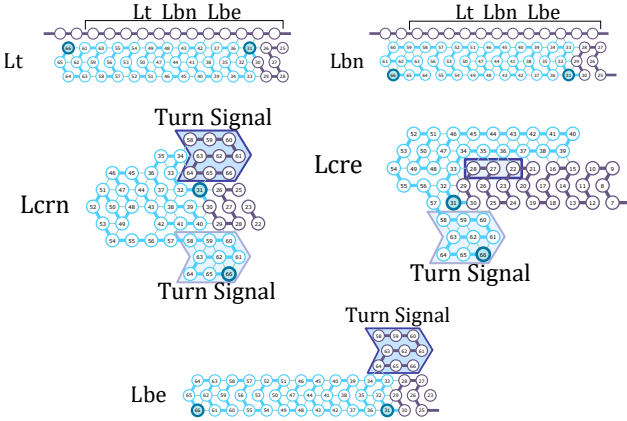

Fig. 5 All the five bricks of module L: Lt, Lbn, Lcrn, Lcre, and Lbe from top left to bottom right. In zigs, L folds into either Lt or Lbn depending on where it starts, until the transcript reaches the left end, where L folds either into Lcrn if the current value has not been overflowed, or into Lbe at an overflow. In the case of overflow, the next L folds into Lcre. In zags, L always folds into Lbn.
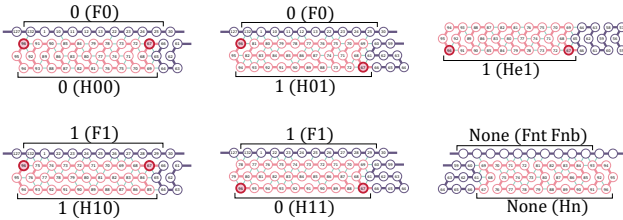
Fig. 6 All the six bricks of module H: H00, H01, He1, H10, H11, and Hn from top left to bottom right. In zags, H always folds into Hn while in zigs, it folds into one of the other five bricks.

4 Conclusion

In this work, we have implemented an infinite binary counter in oritatami system by improving the existing binary counter in order to handle an overflow. We divide the transcript of the infinite binary counter into four modules. The counter is folded into zig-zag manner and both of zig and zag have $4n-3$ modules in order for module L and module R to left turn and to right turn, respectively. The transcript of the existing finite counter has to bind to other modules, on the other
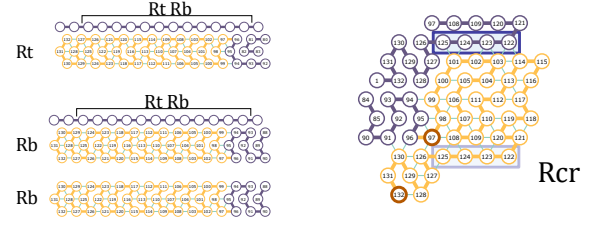

Fig. 7 All the three bricks of module R: Rt, Rb, and Rcr. In zigs, R folds into Rt or Rb, depending on how high it starts. In zags, R always folds into Rb until the transcript reaches the right end, where R folds into Rcr.
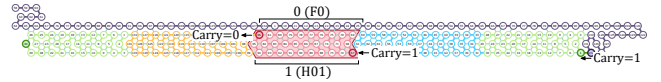

Fig. 8 The first zig. 0 is encoded as an initial value below the seed in the format (2) with $k = 1$ (1-bit width). Being fed with carry, the zig increments the value. Module H outputs 1, or more precisely a sequence of bead types which shall be interpreted as 1 in the next zag and reformatted, as a sum and cancels the carry.
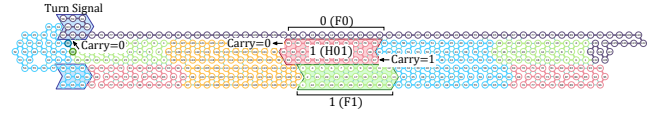

Fig. 9 Module L turns and start the first zag pass. Since there is a Turn Signal at the left end of the seed, when the carry is 0, module L turns and at the end of L forms Turn Signal. In zag pass, module F reads the output of module H and copies it down,
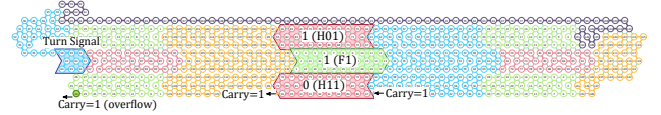

Fig. 10 Reach the left end with carry. Even if transcript of module L sticks to Turn Signal, it can not bind because the distance is long.

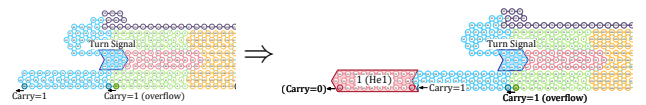hand, our transcript of the infinite counter can fold into a module by inner bounds.

Fig. 11 (Left) Starting from the bottom, the Turn Signal above is too far for this L to fold into Lcrn. It rather folds into Lbe and initiates bit expansion. (Right) Without anything around, the succeeding H folds into a glider (brick He1).
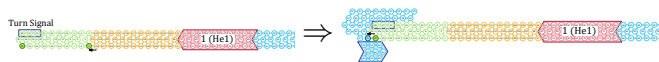
Fig. 12 The succeeding R and F also fold into respective glider-like bricks. (Left) This brick of F (`Fnb`) exposes Turn Signal **28-27-22** (boxed), which is usually "hidden" under the previous zag. (Right) The exposed **28-27-22** (boxed) triggers the folding of next L into a special brick (`Lcre`) for left carriage-return.

References

1. Adleman, L., Chang, Q., Goel, A., Huang, M.D.: Running time and program size for self-assembled squares. In: Proc. STOC 2001, ACM. pp. 740–748 (2001)
2. Bryans, N., Chiniforooshan, E., Doty, D., Kari, L., Seki, S.: The power of nondeterminism in self-assembly. Theory of Computing 9, 1–29 (2013)
3. Evans, C.G.: Crystals that Count! Physical Principles and Experimental Investigations of DNA Tile Self-Assembly. Ph.D. thesis, Caltech (2014)
4. Geary, C., Étienne Meunier, P., Schabanel, N., Seki, S.: Proving the Turing universality of oritatami cotranscriptional folding. In: Proc. ISAAC 2018. pp. 23:1–23:13 (2018)
5. Geary, C., Étienne Meunier, P., Schabanel, N., Seki, S.: Oritatami: A computational model for molecular co-transcriptional folding. Int. J. Mol. Sci. 20(9), 2259 (2019)
6. 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)
7. Masuda, Y., Seki, S., Ubukata, Y.: Towards the algorithmic molecular self-assembly of fractals by cotranscriptional folding. In: Proc. CIAA 2018, LNCS 10977. pp. 261–273. Springer (2018)
8. McClung, C.R.: Plant circadian rhythms. The Plant Cell 18, 792–803 (2006)
9. Minsky, M. (ed.): Computation: Finite and Infinite Machines. Prentice-Hall, Inc. (1967)
10. Rothemund, P.W.K., Winfree, E.: The program-size complexity of self-assembled squares (extended abstract). In: Proc. STOC 2000, ACM. pp. 459–468 (2000)