Simple Intrinsic Simulation of Cellular Automata in Oritatami Molecular Folding Model

Daria Pchelina[1], Nicolas Schabanel[2⋆], Shinnosuke Seki[3⋆⋆], and Yuki Ubukata[4]

[1] École Normale Supérieure de Paris, France
[2] École Normale Supérieure de Lyon (LIP UMR5668, MC2, ENS de Lyon), France
[3] U. of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo, 1828585, Japan.
[4] NTT DATA Corporation, Tokyo, Japan

Abstract. The Oritatami model was introduced by Geary et al (2016) to study the computational potential of RNA cotranscriptional folding as first shown in wet-lab experiments by Geary et al (Science 2014). In the Oritatami model, a molecule grows component by component (named beads) into the triangular grid and folds as it grows. More precisely, the δ last nascent beads are free to move and adopt the positions that maximize the number of bonds with the current folded structure. Geary et al (2018) proved that the Oritatami model is capable of efficient Turing universal computation using a complicated construction that simulates Turing machines via tag systems. We propose here a simple Oritatami system which intrinsically simulates arbitrary 1D cellular automata. Being intrinsic, our simulation emulates the behavior of cellular automata in a readable way and in time linear in space and time of the simulated automaton. The Oritatami model has proven to be a fruitful framework to study molecular reconfigurability. Our construction relies on the development of new mechanisms which are simple enough that we believe that some simplification of them may be implemented in the wet lab. An implementation of our construction can be downloaded for testing.

Keywords: Molecular Self-assembly · Co-transcriptional folding · Intrinsic universality · Cellular automata · Turing universality

1 Introduction

DNA computing encompasses the field which tries to implement computation at the molecular levels. A recent example is [17], which implements arbitrary 6-bit cellular automata onto DNA nanotubes, realising a first DNA-based universal computer (limited to 6 bits of memory). This success of the field was built by going back and forth between theory, models and experiments. The Oritatami

⋆ His work is supported in part by the CNRS grants MOPREXPROGMOL and AMARP from the Mission pour l'interdisciplinarité

⋆⋆ His work is supported in part by JSPS KAKENHI Grant-in-Aids for Challenging Research (Exploratory) No. 18K19779 and JST Program to Disseminate Tenure Tracking System, MEXT, Japan, No. 6F36.

model was introduced in 2016 by [4] to study the computational potential of RNA cotranscriptional folding as first shown in wet-lab experiments by [5].

In Oritatami systems, we consider a finite set of *bead types*, and a periodic sequence of *beads*, each of a specific bead type. Beads are attracted to each other according to a fixed symmetric relation. In any folding (*configuration*), a *bond* is formed between any pair of beads located at adjacent positions and attracting each other. At each step, the latest few beads in the sequence are allowed to explore all possible positions, and adopt only those positions that minimise the energy, or otherwise put, those positions that maximise the number of bonds in the folding. "Beads" are a metaphor for domains, i.e. subsequences, in RNA and DNA (and are thus not limited to 4 types only). The Oritatami model has proven to be a fruitful framework to study *molecular reconfigurability*, one of the most promising directions to reduce error in wetlab molecular implementation as error might be erased by reconfiguration later on. Indeed, programming Oritatami systems consists of designing molecules whose shape changes depending on their contexts, hence achieving some form of reconfiguration. Other models studying molecular reconfiguration include nubots [16] and signal passing tile assembly [10, 11]. Previous work on Oritatami includes among others the implementation of a binary counter [4], the Heighway dragon fractal [7], folding of shapes at small scale [2], NP-hardness of the rule minimization [6, 9], a study of its parameters [13], and polynomial-time Turing machine simulation [3].

Our contribution. The universality result by Geary et al. in [3] relies on a complicated construction that simulates Turing machines via tag systems [1, 18]. We propose here a simple Oritatami system which intrinsically simulates arbitrary 1D cellular automata. Being intrinsic [8, 15], our simulation emulates the behavior of cellular automata in a readable way and in time which is linear in the space and time of the simulated automaton. Precisely, our main result is:

Theorem 1 (Main result). *There is a universal finite set of 183 bead types $\mathcal{B}$ such that for any 1D cellular automaton A with Q states and radius r, there is a delay-2 Oritatami system with bead types in $\mathcal{B}$ and periodic transcript with period precisely*

$$\frac{71}{3}\left((3+q)\cdot 2(Q_r)^2 + 8(2q \mod 3)\right) + 10q + 610 \sim \frac{142}{3}(Q_r)^2 \log_2 Q_r$$

that simulates A intrinsically with a supercell shaped as a lozenge with sides of size $O((Q_r)^2 \log Q_r)$, where $q = \lceil \log_2(2Q^{2r+1}) \rceil$ and $Q_r = 2^q \leqslant 4Q^{2r+1}$.

This improves the previous construction in [3] as the number of bead types is only 183 (instead of 542) and the delay is 2 (instead of 3). Furthermore, our construction relies on the development of new mechanisms which are now simple enough to believe that some simplification of them may be implemented in the wet lab.

An implementation of our construction can be downloaded for testing [14].

2 Model and Preliminary results

2.1 Oritatami model

Let B be a finite set of *bead types*. A *configuration* c of a bead type sequence $p \in B^* \cup B^{\mathbb{N}}$ is a directed self-avoiding path $c_0c_1c_2\cdots$ in the triangular lattice $\mathbb{T}$,[5] where for all integer i, the vertex c_i of c is labeled by p_i and refers to the *position* in $\mathbb{T}$ of the $(i+1)$-th bead in the configuration. A *partial configuration* of p is a configuration of a prefix of p. The class of all the configurations obtained by applying an isometry of $\mathbb{T}$ to a given configuration is called a *conformation*.

For any partial configuration c of some sequence p, an *elongation* of c by k beads (or *k-elongation*) is a partial configuration of p of length $|c|+k$ extending by k positions the self-avoiding path of c. We denote by $\mathcal{C}_p$ the set of all partial configurations of p (the index p will be omitted whenever it is clear from the context). We denote by $c^{\triangleright k}$ the set of all k-elongations of a partial configuration c of sequence p.

Oritatami systems. An *oritatami system* $\mathcal{O} = (p, \heartsuit, \delta)$ is composed of (1) a (possibly infinite) bead type sequence p, called the *transcript*, (2) an *attraction rule*, which is a symmetric relation $\heartsuit \subseteq B^2$, and (3) a parameter δ called the *delay*. $\mathcal{O}$ is said *periodic* if p is infinite and periodic. Periodicity ensures that the "program" p embedded in the oritatami system is finite (does not hardcode any specific behavior) and at the same time allows arbitrarily long computation.[6]

We say that two bead types a and b *attract* each other when $a \heartsuit b$. Furthermore, given a (partial) configuration c of a bead type sequence q, we say that there is a *bond* between two adjacent positions c_i and c_j of c in $\mathbb{T}$ if $q_i \heartsuit q_j$ and $|i-j| > 1$. The *number of bonds* of configuration c of q is denoted by $H(c) = |\{(i,j) : c_i \sim c_j,\ j > i+1,\ \text{and } q_i \heartsuit q_j\}|$.

Oritatami dynamics. The folding of an oritatami system is controlled by the delay δ. Informally, the configuration grows from a *seed configuration* (the input), one bead at a time. This new bead adopts the position(s) that maximize(s) the potential number of bonds the configuration can make when elongated by δ beads in total. This dynamics is *oblivious* as it keeps no memory of the previously preferred positions [3].

Formally, given an Oritatami system $\mathcal{O} = (p, \heartsuit, \delta)$ and a *seed configuration* σ of a *seed bead type sequence* s, we denote by $\mathcal{C}_{\sigma,p}$ the set of all partial configurations of the sequence $s \cdot p$ elongating the seed configuration σ. The considered *dynamics* $\mathscr{D} : 2^{\mathcal{C}_{\sigma,p}} \to 2^{\mathcal{C}_{\sigma,p}}$ maps every subset S of partial configurations of length ℓ elongating σ of the sequence $s \cdot p$ to the subset $\mathscr{D}(S)$ of partial

[5] The triangular lattice is defined as $\mathbb{T} = (\mathbb{Z}^2, \sim)$, where $(x,y) \sim (u,v)$ if and only if $(u,v) \in \cup_{\epsilon=\pm1}\{(x+\epsilon, y), (x, y+\epsilon), (x+\epsilon, y+\epsilon)\}$. Every position (x,y) in $\mathbb{T}$ is mapped in the euclidean plane to $x \cdot \vec{\mathsf{e}} + y \cdot \overrightarrow{\mathsf{sw}}$ using the vector basis $\vec{\mathsf{e}} = (1,0)$ and $\overrightarrow{\mathsf{sw}} = \text{RotateClockwise}\,(\vec{\mathsf{e}}, 120^\circ) = (-\frac{1}{2}, -\frac{\sqrt{3}}{2})$.

[6] Note that we do not impose here a maximal number of bonds per bead (called arity).

configurations of length $\ell + 1$ of $s \cdot p$ as follows:

$$\mathscr{D}(S) = \bigcup_{c \in S} \operatorname*{arg\,max}_{\gamma \in c^{\triangleright 1}} \left(\max_{\eta \in \gamma^{\triangleright(\delta-1)}} H(\eta) \right)$$

The possible configurations at time t of the oritatami system $\mathcal{O}$ are the elongations of the seed configuration σ by t beads in the set $\mathscr{D}^t(\{\sigma\})$.

We say that the Oritatami system is *deterministic* if at all time t, $\mathscr{D}^t(\{\sigma\})$ is either a singleton or the empty set. In this case, we denote by c^t the configuration at time t, such that: $c^0 = \sigma$ and $\mathscr{D}^t(\{\sigma\}) = \{c^t\}$ for all $t > 0$; we say that the partial configuration c^t *folds (co-transcriptionally) into* the partial configuration c^{t+1} deterministically. In this case, at time t, the $(t+1)$-th bead of p is placed in c^{t+1} at the position that maximises the number of bonds that can be made in a δ-elongation of c^t.

2.2 Sweeping 2-fan-in 2-fan-out cellular automata

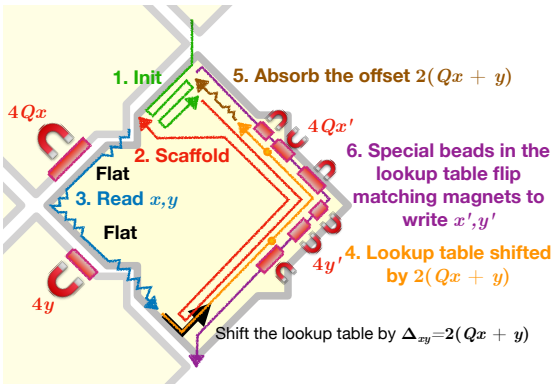
Our construction simulates intrinsically the space-time diagrams of a specific type of one-way cellular automata where each cell has fan-in and fan-out 2 as shown in Fig. 2, similar to the gates implemented in [17]. Formally, a *2-fan-in 2-fan-out automaton (2FA)* $\mathcal{A}$ is given by its set of states $[Q] = \{0, \ldots, Q-1\}$ and its transition function $f : [Q]^2 \to [Q]^2$. A finite configuration of $\mathcal{A}$ is an even-length word $c \in [Q]^*$, and its image by $\mathcal{A}$ is $c' = F(c)$ where $(c'_{2i}, c'_{2i+1}) = f(c_{2i-1}, c_{2i})$ for $i = 0..\frac{|c|}{2} - 1$, with the convention that $c_{-1} = c_{|c|} = 0$. Classically, any 1D cellular automaton with Q states and radius r can be simulated intrinsically by a 2FA with Q^{2r+1} states using a time rescaling by r.

Sweeping simulation. Our construction simulates intrinsically any 2FA by sweeping down (even time step) and up (odd time step), see Fig. 2. As a consequence, every other step, the two inputs are read in reverse order and the transition function is applied with its arguments exchanged. Formally a configuration (c, d) of a *sweeping 2FA (S2FA)* $([Q], f)$ consists of an even-length word $c \in [Q]^*$ together with a direction $d \in \{\uparrow, \downarrow\}$, and has the following dynamics: $F(c, \downarrow) = (c', \uparrow)$ where $(c'_{2i}, c'_{2i+1}) = f(c_{2i-1}, c_{2i})$ for $i = 0..\frac{|c|}{2} - 1$; $F(c, \uparrow) = (c', \downarrow)$ where $(c'_{2i+1}, c'_{2i}) = f(c_{2i}, c_{2i-1})$ for $i = 0..|c|/2 - 1$. Clearly, any 2FA $([Q], f)$ can be simulated intrinsically in real time by the S2FA $([Q] \times \{\uparrow, \downarrow\}, g)$ where $g((x, \uparrow), (y, \uparrow)) = ((x', \downarrow), (y', \downarrow))$ with $(x', y') = f(x, y)$; and $g((x, \downarrow), (y, \downarrow)) = ((x', \uparrow), (y', \uparrow))$ with $(y', x') = f(y, x)$.

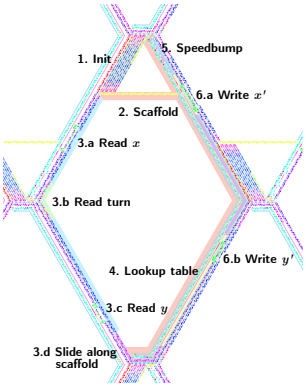
From now on, we consider a S2FA $\mathcal{A} = ([Q], f)$, where $Q = 2^q$ is a power of two with $q \geqslant 1$. We will denote by $(x'(x, y), y'(x, y))$ the value of $f(x, y)$.

3 Overview of the construction

Due to space constraint, we will expose here the principle of the construction. The full description of the modules and of the attraction rule is given in the full version of the present article [12].



(a) Blueprint of the Oritatami cell design



(b) Actual oritatami cell for $q = 1$ ($Q = 2$).

Fig. 1: The modules inside a cell: (Left) Schematic view; (Right) 1. Cell Init highlighted in yellow; 2. Scaffold in red; 3. Read in blue, green and purple; 4. Lookup Table in yellow and violet; 5. Speedbump in cyan; 6. Write in green.

In our intrinsic simulation, each cell of the simulated S2FA is affinely mapped onto a supercell shaped as a hexagon with two short sides (N and S) of lengths 12 and 13, and four long sides (NE, NW, SE and SW) of lengths s and $s-1$ where $s = O(Q^2 \log Q)$ (see Fig. 1b and 2). The states are encoded on the sides of the hexagons as described below. The simulation proceeds by building one after the other the supercells simulating each of the cells of the simulated S2FA according to the up-down order given in Fig. 2. Each supercell is the result of the folding of exactly one period of the transcript. The period of transcript consists of the sequence of 6 *modules*, each of them achieving one specific task:

$$\boxed{\mathsf{I}} \cdot \boxed{\mathsf{S}} \cdot \boxed{\mathsf{R}} \cdot \boxed{\mathsf{L}} \cdot \boxed{\mathsf{SB}} \cdot \boxed{\mathsf{W}}$$

The modules. Their respective roles and positions inside the supercell are blueprinted in Fig. 1a. I is responsible for extending the configuration by one supercell and reversing the up-down order at the end of the current column of supercells (see [12]). S has two roles: providing a scaffold along which the next modules will fold, and ensuring that the molecule "resynchronizes" (will be defined later) before W writes the two outputs x' and y' on the output sides. R is responsible for reading the value of the two inputs x and y and translating accordingly the lookup table of the simulated S2FA, encoded in the next module L. SB is responsible for "resynchronizing" the molecule along the scaffold, annihilating the translation of the lookup table induced by the reading of x and y by R. Finally, W writes on the output sides of the supercell the values x' and y' dictated by the translated lookup table L, and exits the supercell at the entrance of the next one.

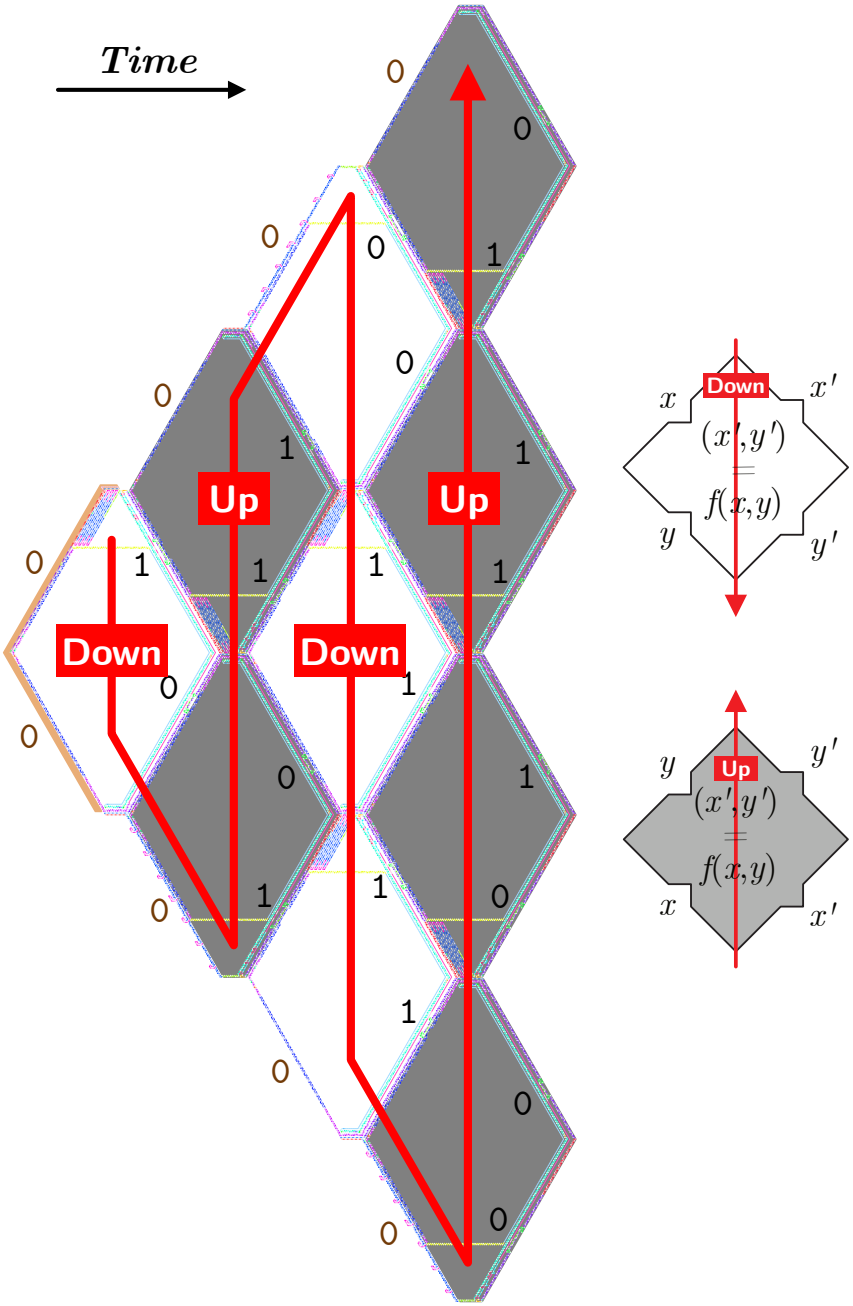


Fig. 2: The 10 first super-steps of the Oritatami simulation of the 2-state S2FA ($q = 1$) $f(x, y) = (y + 1 \bmod 2, x)$ from the seed configuration encoding input 00 (to the left in brown): $(00, \downarrow) \mapsto (0100, \uparrow) \mapsto (011010, \downarrow) \mapsto (00011110, \uparrow) \mapsto (0011110000, \downarrow)$; in white (resp. gray), the down- (resp. up-) hexagonal supercells.

Encoding x and y. The values of x and y are encoded along the sides of the supercells using "magnetic flipping flaps" of total lengths $4Qx$ and $4y$ respectively, as schematically shown on Fig. 1a. When the read module R folds along the side of the neighboring supercells, it gets flattened by these magnetic flaps; this shifts the progression of the molecule forward by exactly half the lengths of the flaps. It follows that the module R completes its folding $\Delta_{xy} = 2(Qx + y)$ beads further than it would in absence of the magnetic flaps. This, in turn, translates the position of the lookup table module L by Δ_{xy} along both output sides of the supercell, placing the entries corresponding to $x'(\Delta_{xy}) = x'(x, y)$ and $y'(\Delta_{xy}) = y'(x, y)$ in front of the flipping flaps of the module W to be folded next so that, when folded, the total magnetic length of the flipping flaps on each output side is $4Qx'$ and $4y'$ respectively (see Section 4 and Fig. 4 for details).

4 Description of the key mechanisms

Due to space constraints, we will focus on the new mechanisms involved in this construction. In particular, we will not discuss I because its behavior is just a direct translation of the Module G in [3] (see [12] for details). S is simply hardcoded and only its key part will be discussed next in Section 4.2.

4.1 Modules R, L, and W: The read, lookup, write mechanism

The previous section gave the principle of the interactions between these modules: the reading of x and y on the input sides by R results in shifting the lookup table L by $\Delta_{xy} = 2(Qx + y)$, which aligns the entries corresponding to $x'(x, y)$ and $y'(x, y)$ properly with the flaps of module W which, in turn, writes the corresponding $x'(x, y)$ and $y'(x, y)$ on the x'- and y'-output sides respectively using the magnetic flaps as illustrated in Fig. 4. Let us start with Module L. Refer to Fig. 3 for the alignment of the various parts involved.

Module L. Each output $x'(x, y)$ and $y'(x, y)$ is encoded in binary into q tables of Q^2 bits using bead types **Q0** and **Q1**. The entry indexed $Qx+y$ in the i-th table for x' (resp. y') contains the value of the i-th bit of $x'(x, y)$ (resp. $y'(x, y)$). More precisely, if we write $x'(x, y) = \sum_{i=0}^{q-1} b_i 2^i$ in binary, the table for x' consists of the sequence of bead types: $\mathbf{Lookup}_X = \left(\prod_{i=0}^{q-1} \prod_{x=0}^{Q-1} \prod_{y=0}^{Q-1} (\mathbf{Q}(b_i))^2\right)^R$, such that the bead types in $\mathbf{Lookup}_X{}^R$ at indices $0, 2Q^2, \ldots, (q-1)2Q^2$ shifted by $\Delta_{xy} = 2(Qx + y)$ are $\mathbf{Q}(b_0), \ldots, \mathbf{Q}(b_{q-1})$. $\mathbf{Lookup}_Y$ is defined similarly.

Module W consists of a zigzag glider **T0..7** that runs along the two output sides of the supercell, together with q "magnetic flipping flaps," equally spaced by $2Q^2$ beads on each output side (see Fig. 3): q flaps of lengths $2^0Q, \ldots, 2^{q-1}Q$ on the x'-output side and of lengths $2^0, \ldots, 2^{q-1}$ on the y'-output side. We define a *magnetic flipping flap of length ℓ* as the bead type sequence:

$$\mathbf{SegFF}(\ell) = \mathbf{U0..5}\,(\mathbf{T4}\,\mathbf{P4}\,(\mathbf{T6}\,\mathbf{P0}\,\mathbf{T0}\,\mathbf{P3}\,\mathbf{T2}\,\mathbf{P2}\,\mathbf{T4}\,\mathbf{P1})^{\ell}\mathbf{T6}\,\mathbf{P0..4}\,\mathbf{U6..8}.$$

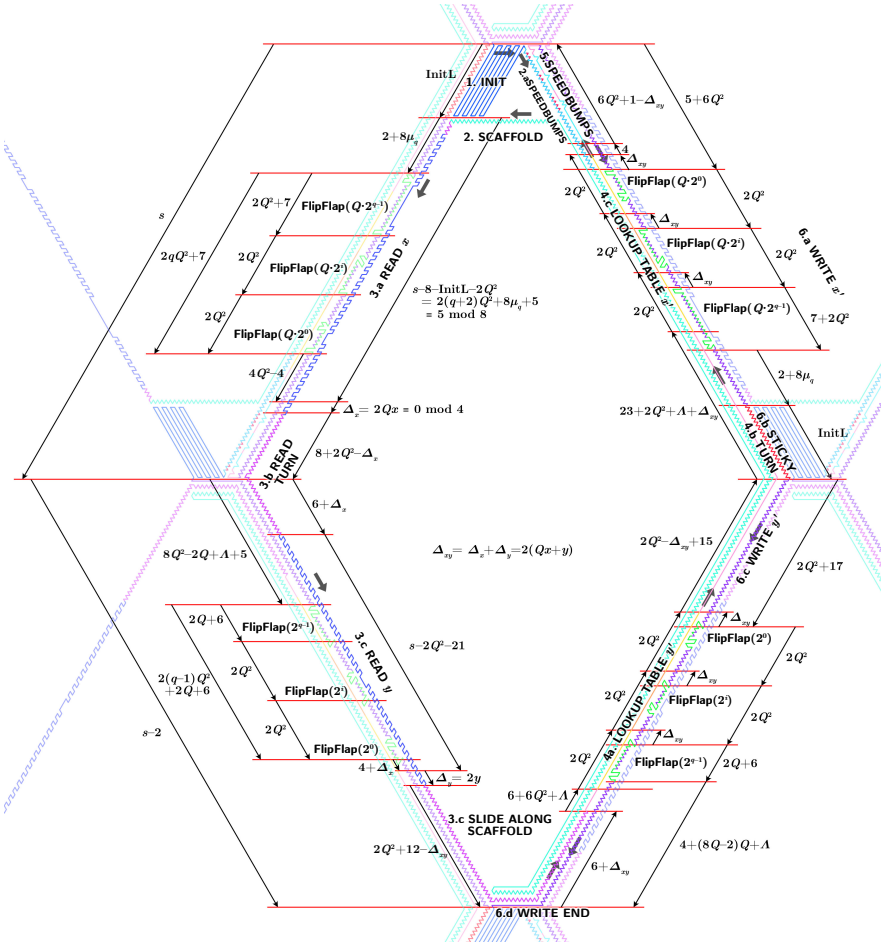
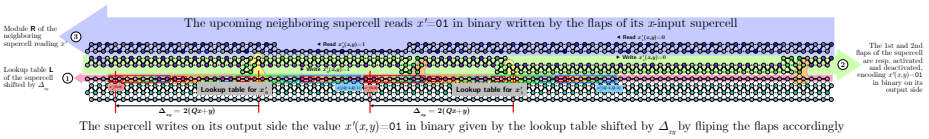


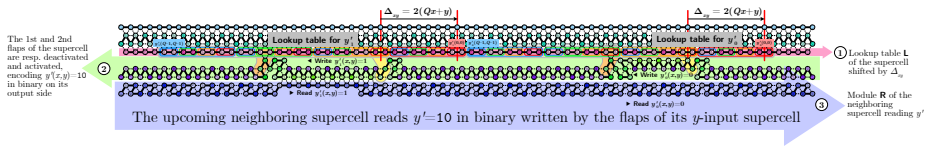
Fig. 3: Alignment of the various modules

Each flap is either activated (magnetic for R) or deactivated (neutral for R) depending on whether its "magnetic" beads **P0..3** point *outwards*, towards the upcoming neighboring supercell, or *inwards*, towards the inside of the supercell currently folding (see Fig. 4). Now, thanks to the alignment of the modules (see Fig. 3), the i-th flap of W starts folding in front of the entries $\Delta_{xy} + 2iQ^2$ of the lookup table on each output side, that is in front of the pair of beads $\mathbf{Q0}^2$ or $\mathbf{Q1}^2$ corresponding to the i-th bit of the value to write on this side. Now, a flap folds outwards (is activated) by default, unless its initial bead **U5** is attracted by a pair of beads **Q0** corresponding to a bit set to `0`. It follows that the i-th flap of W on each side is activated if and only if the i-th of the output is `1`; and as it is of length 2^iQ and 2^i for the x'- and y'-output side respectively, the

total numbers of magnetic beads are $4Qx'(x,y)$ and $4y'(x,y)$ on each x'- and y'-output side, respectively.



(a) *1) Lookup table* L*, 2) Write* W *and 3) Read* R *modules* on the x'-output side when offset $\Delta_{xy} = 2 \times (4 \times 2 + 1)$ (i.e. $(x, y) = (2, 1)$)



(b) *1) Lookup table* L*, 2) Write* W *and 3) Read* R *modules* on the y'-output side when offset $\Delta_{xy} = 2 \times (4 \times 1 + 0)$ (i.e. $(x, y) = (1, 0)$)

Fig. 4: *Illustration of the border between two neighboring supercells:* Interactions of 1) the lookup table and 2) the write modules within a supercell and 3) the read module of the upcoming neighboring supercell, when $Q = 4$ ($q = 2$). The lookup tables for each bit follow each other and are $2Q^2$ beads long each (2 beads per bit). The write module folds into $q = 2$ flipping flaps on each output side of lengths 4×2^0 and 4×2^1 on the x'-output side and 2^0 and 2^1 on the y'-output side. We have highlighted in yellow the folding of bead **U5**, which decides the orientation of each flap (in- or out-wards if **Q0** or **Q1** is present resp.). We have highlighted in orange the folding of bead **U6**, which is attracted by all bead types **Q0..2** and restores the orientation of the write glider to defaults after each flap.

Module R. We are now ready to conclude this mechanism by observing that the read module R folds along the write modules of the two neighboring input supercells, and that it gets flattened each time it folds along an activated flap (see Fig. 3 and 4), which extends its length by half the number of magnetic beads **P0..3** of the flap. It follows that the end of its folding is shifted forward by $(4Qx + 4y)/2 = \Delta_{xy}$, which in turn shifts forward the lookup table module L by Δ_{xy} as claimed. Refer to Fig. 6 for a complete view of the folding of R.

The full description of the modules R, L and W may be found in the full version [12] of the present article.

4.2 Modules SB and S: resynchronization using speedbumps

In order for the period of the transcript to end precisely at the exit of the supercell, regardless of which inputs x and y were read by the read module R,

we need to absorb the Δ_{xy} offset. Precisely, we need to absorb it before the write module W folds to ensure that it is properly aligned with the shifted lookup table. This is the role of the speedbump module SB. Its behavior is illustrated in Fig. 5.

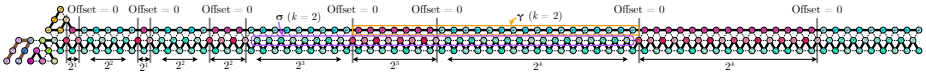
This mechanism involves two modules: the scaffold module S, which contains the speedbumps (consisting of alternation of red beads **I0..3** and blue beads **E0..3**) at the top of its NE corner (assuming the supercell is in the downwards orientation); the speedbump module SB which consists of a matching alternation of red beads **Q2** and blue beads **R0..1**.

Lemma 1 (speedbump). *When a red-blue sequence* $\gamma = \mathbf{Q2}^{4k-1}(\mathbf{R0..1})^{4k}\mathbf{R0}$ *folds from right to left over a blue-red-blue seed left-to-right sequence* $\sigma = (\mathbf{E2E4E6E0})^k(\mathbf{I1..3I0})^k(\mathbf{E2E4E6E0})^{2k}$ *starting from the* Δ*-th rightmost position of* σ *with* $\Delta < 4k$*, the* Δ *leftmost blue beads of* γ *fold into a zigzag over the red beads of* σ*, and the folding of* γ *ends at the* $\lfloor \Delta/2 \rfloor$ *rightmost position of the left red segment of* σ*, as shown in Fig. 5b.*

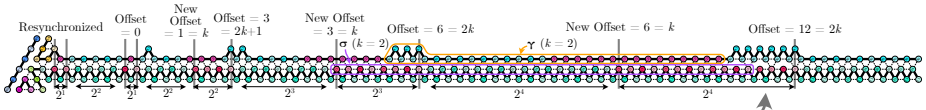
Corollary 1. *When the folding of the speedbump module* SB *completes, the offset* Δ_{xy} *is totally absorbed.*

Proof (sketch). Note that the maximum offset when the speedbump module SB starts to fold is $\Delta = 2(Q^2 - 1)$ corresponding to reading input $(x, y) = (Q - 1, Q - 1)$. The matching exponentially decreasing alternation of blue and red regions from 2^{2q} to 4 in S and SB (see [12]) ensures that the offset is divided by 2 until it reaches 0, absorbing the total offset as shown in Fig. 5.

Correctness of the folding. Finally, the correctness of the folding is proved by induction using automated folding tree certificates (see [4, 12]). The key is to choose carefully the size s of the supercell so that all modules are properly aligned regardless of the inputs x and y. This is ensured by enforcing the position of every pattern in every module modulo 8 in the supercell as explained in Fig. 3 and detailed in [12].



(a) *Synchronized speedbump:* all the top red parts are inside the red speedbumps



(b) *Synchronizing speedbump:* each time the blue part of the molecule passes over a red speedbump, the offset Δ is divided by 2.

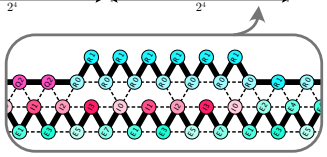


Fig. 5: Speedbumps decrease exponentially the offset of the molecule folding on top (going from right to left) until it vanishes.

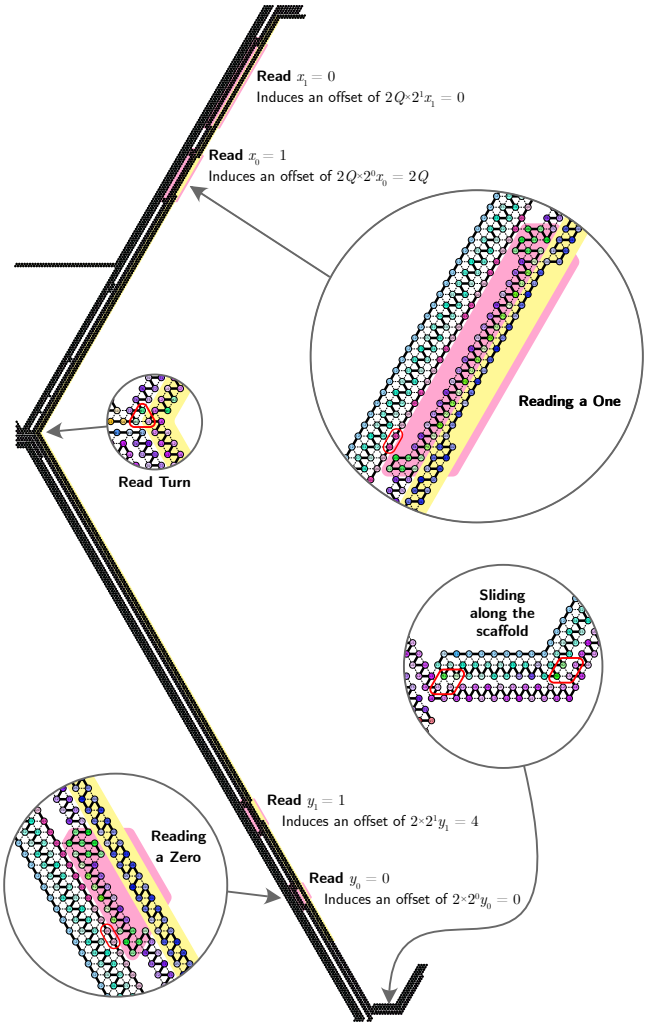


Fig. 6: *Read* module consists of 4 parts: ReadX, Turn, ReadY, and Slide along the scaffold. ReadX and ReadY get flattened each time it passes along an outward write magnetic flap (encoding a 1). This shifts the molecule by the length of the corresponding flap. The following part of the molecule is then shifted overall by $\Delta_{xy} = 2(Qx + y)$. This shift allows then to align the entry corresponding to (x, y) of the lookup table of each side with the upcoming write modules. The two glider-based parts "Turn" and "Slide along" are only there to ensure that after reading each input, the molecule turns at the expected position, regardless of the offset.

References

1. Cook, M.: Universality in elementary cellular automata. Complex Systems **15**(1) (2004)
2. Demaine, E., Hendricks, J., Olsen, M., Patitz, M., Rogers, T., Schabanel, N., Seki, S., Thomas, H.: Know when to fold 'em: Self-assembly of shapes by folding in oritatami. In: DNA. LNCS, vol. 11145, pp. 19–36 (2018)
3. Geary, C., Meunier, P.E., Schabanel, N., Seki, S.: Proving the Turing universality of oritatami cotranscriptional folding. In: ISAAC. LIPIcs, vol. 123, pp. 23:1–23:13 (2018)
4. Geary, C., Meunier, P.E., Schabanel, N., Seki, S.: Oritatami: A computational model for molecular co-transcriptional folding. Int. J. Mol. Sci. **20**(9), 2259 (2019), preliminary version published in MFCS 2016
5. Geary, C., Rothemund, P.W.K., Andersen, E.S.: A single-stranded architecture for cotranscriptional folding of RNA nanostructures. Science **345**, 799–804 (2014)
6. Han, Y.S., Kim, H.: Ruleset optimization on isomorphic oritatami systems. Theor. Comput. Sci. **785**, 128–139 (2019)
7. Masuda, Y., Seki, S., Ubukata, Y.: Towards the algorithmic molecular self-assembly of fractals by cotranscriptional folding. In: CIAA. LNCS, vol. 10977, pp. 261–273. Springer (2018)
8. Ollinger, N.: Two-states bilinear intrinsically universal cellular automata. In: FCT. LNCS, vol. 2138, pp. 396–399. Springer (2001)
9. Ota, M., Seki, S.: Ruleset design problems for oritatami systems. Theor. Comput. Sci. **671**, 26–35 (2017)
10. Padilla, J.E., Patitz, M.J., Schweller, R.T., Seeman, N.C., Summers, S.M., Zhong, X.: Asynchronous signal passing for tile self-assembly: Fuel efficient computation and efficient assembly of shapes. Int. J. Found. Comput. Sci. **25**(4), 459–488 (2014)
11. Padilla, J.E., Sha, R., Kristiansen, M., Chen, J., Jonoska, N., Seeman, N.C.: A signal-passing DNA strand exchange mechanism for active self-assembly of DNA nanostructures. Angew. Chem. Int. Edit. **54**(20), 5939–5942 (2015)
12. Pchelina, D., Schabanel, N., Seki, S., Ubukata, Y.: Simple Intrinsic Simulation of Cellular Automata in Oritatami Molecular Folding Model (Dec 2019), https://hal.archives-ouvertes.fr/hal-02410874, full version of the present article.
13. Rogers, T.A., Seki, S.: Oritatami system; a survey and the impossibility of simple simulation at small delays. Fund. Inform. **154**(1-4), 359–372 (2017)
14. Schabanel, N.: iOS CAOS simulator, hub.darcs.net/nikaoOoOoO/CAOSSimulator
15. Theyssier, G.: Automates Cellulaires: un Modèle de Complexités. Ph.D. thesis, École Normale Supérieure de Lyon (2005)
16. Woods, D., Chen, H., Goodfriend, S., Dabby, N., Winfree, E., Yin, P.: Active self-assembly of algorithmic shapes and patterns in polylogarithmic time. In: ITCS. pp. 353–354 (2013)
17. Woods, D., Doty, D., Myhrvold, C., Hui, J., Zhou, F., Yin, P., Winfree, E.: Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. Nature **567**, 366–372 (2019)
18. Woods, D., Neary, T.: On the time complexity of 2-tag systems and small universal Turing machines. In: FOCS. pp. 439–448 (2006)