O*k*: A Kinetic Model for Locally Reconfigurable Molecular Systems

Pierre Marcus, Nicolas Schabanel, and Shinnosuke Seki

Abstract Oritatami is a formal model of RNA co-transcriptional folding, in which an RNA sequence (transcript) folds upon itself while being synthesized (transcribed) out of its DNA template. This model is simple enough for further extension and also strong enough to study computational aspects of this phenomenon. Some of the structural motifs designed for Turing universal computations in oritatami have been demonstrated approximately in-vitro recently. This model has yet to take a significant aspect of co-transcriptional folding into full account, that is, reconfiguration of molecules. Here we propose a kinetic extension of this model called the oritatami kinetic (O*k*) model, similar to what kinetic tile assembly model (*k*TAM) is to abstract tile assembly model (*a*TAM). In this extension, local rerouting of the transcript inside a randomly chosen area of parameterized radius competes with the transcription and the folding of the nascent beads (beads are abstract monomers which are the transcription units in oritatami). We compare this extension to a simulation of oritatami in the nubot model, another reconfiguration-based molecular folding model. We show that this new extension matches better a reconfiguration model and is also faster to simulate than passing through a nubot simulation.

1 Introduction

Transcription is a phenomenon in which a system encoded on a DNA sequence is copied sequentially out of ribonucleic acids by an RNA polymerase into an RNA transcript. The particularity of this process is that the transcript folds upon itself into an intricate structure while being synthesized, that is *co-transcriptionally*.

P. Marcus (✉)
École Normale Supérieure de Lyon (LIP, MC2), Lyon, France
e-mail: pierre.marcus@ens-lyon.fr

N. Schabanel (✉)
CNRS, École Normale Supérieure de Lyon (LIP, MC2), Lyon, France
e-mail: nicolas.schabanel@ens-lyon.fr

S. Seki (✉)
University of Electro-Communications, Tokyo, Japan
e-mail: s.seki@uec.ac.jp

N. Jonoska and E. Winfree (eds.), *Visions of DNA Nanotechnology at 40 for the Next 40*,
Natural Computing Series, https://doi.org/10.1007/978-981-19-9891-1_13

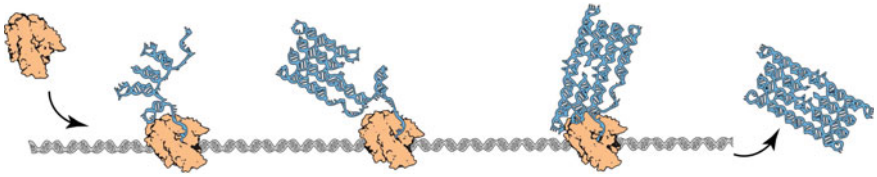

Fig. 1 Transcription and RNA origami [6]: An RNA polymerase (colored in orange) scans a DNA template (gray) and maps its sequence, nucleotide by nucleotide (A, T, C or G), into an RNA sequence (the *transcript*) according to the loss-less function A → U, C → G, G → C, and T → A. The transcript folds upon itself with high probability into a precise structure that can be programmed from the sequence of the DNA template

This phenomenon, called *co-transcriptional folding*, has proven to be programmable in-vitro. Indeed, in [6] Geary, Rothemund and Andersen demonstrated how to encode a rectangular tile-like structure in a transcript (actually, in its corresponding DNA template) so that following its folding pathway, the transcript folds co-transcriptionally into the target structure (Fig. 1). The design of such an *RNA origami* architecture has been highly automated by their software RNA Origami Automated Design (ROAD) [2]. ROAD "extends the scale and functional diversity of RNA scaffolds" so that they might be large and functional enough even to accommodate simple enough computation.

Besides serving as a scaffold for computation, co-transcriptional folding itself is capable of computing by encoding several folding pathways into a single transcript and letting an appropriate one be "called" depending on the environment [8, 9, 12]. The *oritatami* model was introduced in [3] to explore theoretically the computation capabilities allowed by co-transcriptional folding. It was first demonstrated to be capable of counting in binary [3, 7] and then of simulating arbitrary cyclic tag system [5]. As such, the oritatami model is efficiently Turing universal: it can simulate arbitrary Turing machines with a quadratic-time slow down only. Precisely, a cyclic tag system (CTS) is a binary word (over {0, 1}) rewriting system that consists of an initial tape word w^0 and a finite cyclic list of productions (binary words) and yields a sequence of nonempty tape words $w^0, w^1, w^2, \ldots$; at step $i \geq 1$, it rewrites the tape word w^{i-1} into w^i by (1) appending the current production at the end of w^{i-1} if and only if its leftmost letter w_0^{i-1} is 1 (and appending nothing otherwise), and then (2) deleting this first letter, and (3) rotating the cyclic list.

The oritatami CTS simulator [5] encodes the cyclic list of a given CTS in each period of its cyclic (periodic) transcript. The encoded list folds into a compact shape (called *switchbacks*) by default, unless a production must be appended at the end of the current tape word, in which case the current production inside that list folds in a self-supported expanded shape (called *glider*), extending the current tape word accordingly.

Multiple configurations in co-transcriptional molecular folding computing. As computation is achieved in tile assembly systems by gluing tiles together in different configurations in response to its surrounding, computation is achieved in co-

transcriptional folding model by having the transcript be folded in different shapes in response to its environment, i.e., its context. Some of the RNA structures created by ROAD [2] resemble structural motifs used for computing in oritatami, such as glider and switchbacks. That been said, the ability to encode two addressable shapes into one single sequence has not been demonstrated experimentally yet. Such compatibility between two foldable shapes, however, might be dispensable. Indeed, the Turing universal transcript in oritatami has been simplified significantly in [10]. In this later work, the periodic transcript folds co-transcriptionally into a space-time diagram of a 1D cellular automaton (CA). Each period of the transcript folds into a macrocell, an upscaled version of the corresponding simulated CA cell. The macrocell is divided into three parts: (1) the first detects the absence of a neighboring cell and build the missing boundary in that case, (2) the second builds the inner shell of the macrocell, and (3) the third part reads the input bits encoded on its NW and SW borders, and writes the output bits on its SE and NE borders. Only the first part requires a compatibility between two non-trivial patterns (actually, between glider and switchback). The inner shell is indeed hardcoded in the second part and uses only gliders and 60°- and 120°-turns. The third part consists of, first, reading gliders that get locally flatten when facing beads of specific types and, second, of a flat line encoding a transition table. The reading gliders get flat when passing by the parts of a neighboring macrocell border encoding a `1`, which shifts forward the upcoming transition by an appropriate amount so as to expose only the output entry corresponding to the input. An interesting feature of this new I/O interface is its tolerance to misalignment of macrocells, a typical desired outcome in presence of molecular reconfigurations, even though the macrocells never get misaligned in the deterministic oritatami system.

The need for a co-transcriptional reconfiguration model. As opposed to tile assembly model, experimental evidence of computing using molecular co-transcriptional folding has not yet been proven, even though it has been shown theoretically possible thanks to the oritatami model. However, one main obstacle is the lack of a model which would take into account the probabilistic nature of experimental settings. This was solved for the tile assemblies by introducing the kinetic tile assembly model (*k*TAM) which provided useful hints, such as proofreading tiles [13], which allowed in turn to conduct successful experimental implementation of computing nanostructures [1, 14]. Co-transcriptional folding is significantly different from tile assembly as it is not composed of independent entities but requires that all the beads or monomer composing the resulting structure to be connected by a path. The *nubots* model introduced in [14] includes reconfigurations and allows to build structure with this kind of constraint, and even much more complex ones. Even if there is a way to simulate oritatami systems with nubots, this simulation is indirect and passes through unnatural intermediate states that should not be considered in a kinetic model (Sect. 2). This motivates the introduction of a new model, called *Oritatami kinetic* (O*k*) model, which extends the oritatami model to include thermal reconfigurations of the folding structure. We hope this model to be powerful yet simple enough to design co-transcriptional folding scheme that will be robust enough to be imple-

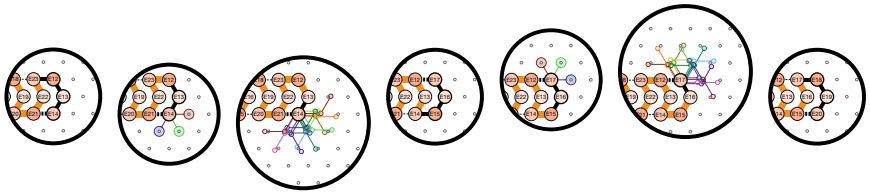

Fig. 2 ***Oritatami model:*** From left to right, the growth from bead `E12` to bead `E18` of a self-supported glider with delay $\delta = 3$, transcript $p =$ `E12` ... `E23` and rule {`E12` ♥ `E17`, `E14` ♥ `E21`, `E18` ♥ `E23`, `E20` ♥ `E15`}. At each step, the set of nascent paths and maximizing the number of bonds is shown. The nascent beads are highlighted in bold black. The nascent paths are drawn in bold black until the last bond made and ends in colors when their tail is free to move (i.e., is not bounded by any bond)

mented in-vitro. Furthermore one might hope that it may also open new way of design co-transcriptionally folding structures taking advantage of these new features (Fig. 2).

2 Molecular Reconfiguration: Oritatami and Nubots

Let us first compare the oritatami and nubots models.

Oritatami model. An oritatami system consists of a "molecule" (the *transcript*) consisting in a sequence of "*beads*" (monomers) that attract each other according to a given binary relation ♥ called the *rule*. The molecule grows in the triangular lattice, by one bead per step. At each step, the δ most recently produced *nascent* beads are free to move around to look for the position that maximizes the number of bonds they can make with each other or with beads placed already (hence the folding is co-transcriptional). Once that optimal position is found, the oldest nascent bead will adopt this position *forever*. Then, a new nascent bead is produced (according to the transcript sequence) and the process continues by optimizing the position of the new δ nascent beads. The transcript, i.e., the sequence of beads, is assumed to be finite or periodic. The parameter δ is called the *delay*. The folding starts from an initial configuration called the *seed*. A time- and space-efficient and easy-to-use oritatami model simulator is freely available at [11].

Nubots model. Nubots is a general purpose model capturing many (if not all) aspects of 2D molecular reconfigurability. Nubots are grown in the triangular lattice as well. They are composed of monomers that interact with their lattice neighbors in a non-deterministic manner. At each time step, a monomer can change its internal state, create a new neighbor monomer, or conversely disappear, change the nature of its bond with one of its neighbors (none, rigid or flexible), or move around one of its bonded neighbors taking along a part of its bond-connected component. Each of these

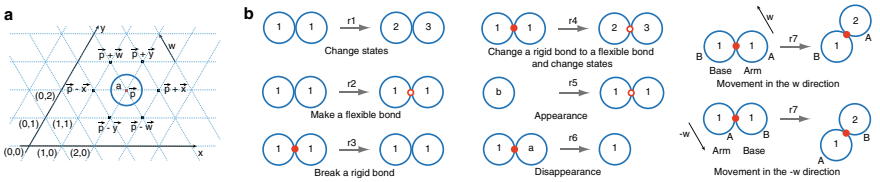

Fig. 3 ***Nubots model***: (Figure extracted from [14]) **a** A monomer in state a at position $\vec{p}$ in the triangular grid coordinate system $(\vec{x}, \vec{y}, \vec{w})$. **b** Examples of monomer interaction rules, written formally as follows: $r_1 = (1, 1, \mathsf{null}, \vec{x}) \to (2, 3, \mathsf{null}, \vec{x})$, $r_2 = (1, 1, \mathsf{null}, \vec{x}) \to (1, 1, \mathsf{flexible}, \vec{x})$, $r_3 = (1, 1, \mathsf{rigid}, \vec{x}) \to (1, 1, \mathsf{null}, \vec{x})$, $r_4 = (1, 1, \mathsf{rigid}, \vec{x}) \to (2, 3, \mathsf{flexible}, \vec{x})$, $r_5 = (b, \mathsf{empty}, \mathsf{null}, \vec{x}) \to (1, 1, \mathsf{flexible}, \vec{x})$, $r_6 = (1, a, \mathsf{rigid}, \vec{x}) \to (1, \mathsf{empty}, \mathsf{null}, \vec{x})$, and $r_7 = (11, \mathsf{rigid}, \vec{x}) \to (1, 2, \mathsf{rigid}, \vec{y})$. For rule r_7, the two potential symmetric movements are shown corresponding to two choices for arm and base, one of which is non-deterministically chosen

possibilities is described in a list of possible actions between any pair of neighboring monomers according to their respective internal states (Fig. 3). As for the oritatami model, the process starts from an initial configuration called the seed. This model takes advantage of both local and parallel reconfigurability to build large structure in a logarithmic number of parallel updates only.

Oritatami and nubots. One approach to introduce reconfigurability in the oritatami model is to implement it in the nubot model.

Theorem 1 *Any oritatami system can be implemented as a nubot.*

Proof Consider an oritatami system O with seed σ, periodic transcript p, rule and delay δ. The simulating nubot will consist of two kind of monomers: placed monomers and nascent monomers. Placed monomers correspond to the beads that have already been placed at their final location; their internal state will be the bead type of the corresponding bead according to the periodic transcript p. The nubots will grow and retract a chain of monomers linked by rigid bonds corresponding to the folded oritatami molecule. Nascent monomers correspond to the δ nascent beads of the simulated oritatami system; their internal state will encode not only the bead type of their corresponding bead but also some finite amount of information allowing to explore one by one all the possible paths, so as to compute the paths that maximize the number of bonds that the nascent beads can make in the simulated oritatami configuration. Note that the simulating nubot will conduct the path exploration by a simple depth first search where each nascent monomer spawn its child nascent monomer in every possible direction in a recursive manner. To conduct this exploration, each nascent monomer only needs to remember the currently explored direction, the current best number of bonds made by its descendants in some already explored directions, and its index in the transcript (to specify its bead type). As the total number of feasible bonds may not exceed $4\delta + 1$ (4 for the intermediate nascent monomer and 5 for the tailing one), the required number of internal states is $O(\log \delta + \log |p|)$, that is constant. Each parent nubot keeps the best number of

bonds among all of its children and adds its own number of bonds with the current environment, and then sends it to its parent before disappearing, until the process reaches back the path root nubot. This root will then be able to extend the oritatami path in the direction of the best move and will then restart the exploration from there.

This simulation of an oritatami system by a nubot is however unsatisfying as it requires the regular disappearance of monomers and thus introduces unnatural, and thus undesirable, intermediate states in the simulation. Other nubots simulation schemes exist at scale 1 that do not require the disappearance rule, but require instead to disconnect the nascent beads to rotate them around the environment, or to make room by pushing the environment around, but both of these intermediate states are equally undesirable.

Claim Nubots cannot mimic the nascent beads moves at scale 1 without the disappearance rule nor disconnecting the nascent beads.

Indeed, consider a seed configuration consisting of an Y-shaped tunnel with two arms of length δ where the oritatami system starts at the intersection. There are exactly two bead types, A and B, and the only attraction rule is A ♥ B. Its transcript is A^δ (A repeated δ times). The wall of the tunnels are made of As but the bead placed at the end of the tunnel can either be A or B. In order to simulate faithfully the oritatami system, the nubot must reach both ends of the tunnel (otherwise one could exchange the two beads at the end of tunnel and invalidate the simulation). As no part of the grown nascent arm can be moved in any direction without being disconnected there are no other possibility than erasing the nubots grown when the wrong tunnel is explored first, which can be guaranteed by placing B in the second tunnel to be explored by the nubots.

3 The O*k* model

The previous section showed that nubots are not well suited to model a dedicated kinetic co-transcriptional folding model because it would require passing through many unrealistic and time-costly intermediate states. Furthermore, in order to get closer to nature, we want, as in kTAM, our kinetic oritatami model to randomly reconfigure parts that have been folded already. As the connectivity of the path (made of covalent bonds) must be maintained, this would require lots of computation steps as well as lots of memory if modeled inside the nubots world.

3.1 Reconfiguration Events

In the *oritatami kinetic (Ok)* model, the dynamics changes. Instead of optimizing the nascent beads positions after each extension of the transcript by one bead, there are three kinds of reconfiguration events that are competing with each other:

Growth: A new bead is added at the end of the growing molecule. Its bead type is given by the transcript. It is placed at a uniformly and randomly chosen unoccupied location next to the current end of the path. If none of these locations is unoccupied, the growth fails and will be retried the next time the growth event is triggered.

Nascent beads reconfiguration: The path followed by the δ nascent beads (the δ most recently produced) is rerouted non-deterministically without overlaps, according to some probabilistic distribution to be discussed next.

Internal reconfiguration: An hexagon H of radius ρ was picked at random and all the subpaths inside this hexagon H are rerouted non-deterministically while keeping their extremities on the borders of H unchanged (see Fig. 4).

Figure 4 gives an example of a reconfiguration. Note that none of the reconfiguration events involves directly any bonding scheme optimisation process, as opposed to regular oritatami model. As for kTAM, this optimization will be induced by the rates at which these various reconfigurations are applied, as will be detailed in the following section.

3.2 Reconfiguration Distributions and Events Rates

Each of the reconfiguration events Growth, Nascent beads, and Internal reconfigurations will be triggered according to exponential random variable of respective rates r_G, r_N and r_I. The growth rate r_G only depends on the transcription speed of the polymerase (which may depends in turn on concentrations, temperature,...). The nascent beads and internal reconfiguration rates, r_N and r_I, do however depend on the local configuration where they are applied: the more bonds are made, the more stable the local configuration is and the less likely reconfiguration is made.

Local reconfiguration random distribution. In the Ok model, we assume that when a reconfiguration (nascent or internal) is applied, the new local configuration (of the nascent path or of the subpaths in the hexagon H) is drawn *uniformly at random* among all the valid local reconfigurations. Together with the upcoming definition of the rates, this ensures that the resulting distribution follows Boltzmann law.

Reconfiguration events rates. Following the steps of [13], we define the rate of each reconfiguration as a function of the number of bonds involved: a configuration involving b bonds will dissociate at a rate inversely proportional to some exponential of b. Precisely, as in [13] we define the *free energy of dissociation* of a single bond to be $\Delta G/RT = G_B$, and thus G_B contains a mix of entropic and enthalpic factors

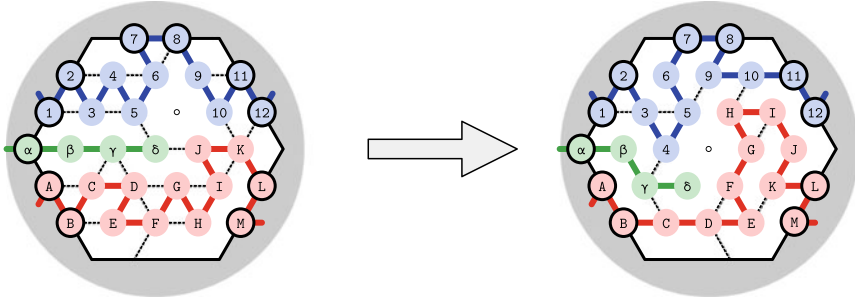

Fig. 4 ***Internal Reconfiguration inside an hexagon*** H ***of radius*** ρ ***(here*** $\rho = 3$): The reconfiguration involves four subpaths: three of them are clamped `2-3-4-5-6-7`, `8-9-10-11` and `B-C-D-E-F-G-H-I-J-K-L`, and one has a free end, $\alpha - \beta - \gamma - \delta$. The reconfiguration step changes their routes inside the hexagon. Beads on the border of H (circled in bold) or outside do not move. The left and right configurations make 20 and 10 bonds (dashed lines), respectively, inside the ball

related to the formation of the bond between the two beads, measured in units of RT. The rate of the next internal reconfiguration event in a given hexagon H is then defined as:

$$r_I = k_f \exp(-bG_B), \tag{1}$$

where b is the number of bonds involving some bead strictly inside the selected hexagon H. k_f is a cinetic parameter, usually set to $k_f = 10^{-6}$.

To take into account the specificity of co-transcriptional folding where the nascent beads, close to the polymerase, fold at a much higher rate, we define similarly the *free energy of dissociation of a single nascent bond* (i.e., involving at least one nascent bead, and thus catalyzed by the proximity to the polymerase) to be $\Delta G/RT = G_N$. The rate of the next nascent reconfiguration event is then defined as:

$$r_N = k_f \exp(-bG_N), \tag{2}$$

where b is the number of nascent bonds, involving at least nascent bead.

In order to express the rate r_G of the growth of the transcript in the same terms, we introduce, as in [13], a fictitious energy G_G such that:

$$r_G = k_f \exp(-G_G). \tag{3}$$

Adjusting the rates. In order to match the oritatami dynamics, the nascent beads must explore up to 5^δ paths, so as to find the optimal nascent path, before the next nascent bead is produced. This exploration requires on average $\sim \delta \ln 5 \cdot 5^\delta$ uniform random nascent reconfiguration events, as collecting N coupons takes on average $N \ln N$ trials, and thus $\sim \delta \ln 5 \cdot 5^\delta / r_N$ time as each event occurs every $1/r_N$ on average. In order to match the co-transcriptional folding experimental observation that the growth occurs at a much lower rate than the optimal folding of the nascent beads, we must then have:

$$\frac{\delta 5^\delta \ln 5}{r_N} \ll \frac{1}{r_G} \quad \text{that is: } r_N \gg \delta 5^\delta \cdot r_G, \text{ i.e.. } bG_N \lesssim G_G - \delta \ln 5 \tag{4}$$

Note that this is consistent with the fact that the number of nascent bonds b is at most $4\delta + 1$: each nascent bond must account for a fixed amount of energy, that is: $G_G \gtrsim \delta(4G_N + \ln 5)$. In particular we should have: $G_G \gtrsim \delta \ln 5/4$ so that $G_N > 0$. This first rough estimation however needs to be confirmed by running effective simulation of the Ok model.

There are no explicit constraints between G_B and G_N besides the fact that $G_B > G_N$ as the nascent bounds should dissociate more easily than the (colder) bonds located away from the polymerase.

3.3 Implementing the Ok model

We do not have a running implementation of the Ok model yet. Here are a list of recommendations for its software implementation.

Ok *model possible implementation.* The Ok model is parametrized by $(\delta, \rho, G_G, G_B, G_N)$. The possible event are:

- growth: extending the molecule by one bead whose position is chosen uniformly at random among the unoccupied positions neighboring the current end of the molecule;
- internal(x): rerouting uniformly at random the subpaths strictly inside the hexagon $H(x, \rho)$ centered on position x with radius ρ, without modifying their extremities on the boundary of $H(x, \rho)$;
- nascent: rerouting uniformly at random the path consisting of the δ nascent beads at the end of the molecule.

There are as many internal(x) events as there are hexagons $H(x, \rho)$ intersecting the molecule. Each event is associated with an occurrence time T picked at random according to their corresponding rate r_G, r_I or r_N. Note that the rates r_I and r_N depend on the current local configuration. As T is a memory-less exponential random variable of law Exp(r), s.t. $\Pr\{T \geq t\} = e^{-rt}$, it can easily be picked using the formula: $T = -(\ln U)/r$ where U is a uniform real-valued random variable over $[0, 1]$. The events scheduling is classically implemented using a priority queue to extract the next upcoming event (with the lowest occurrence time). Thanks to the memory-less property, the occurrence times of all events impacted by an applied event are simply redrawn according to their recomputing rate: for instance, the occurrence times of the internal(y) events need to be updated for all $y \in H(x, \rho)$ after an internal(x) event occurred.

Performance optimization. The implementation of the nascent and internal will get computer intensive as soon as δ and ρ get larger than 10 and 5 respectively. Even for smaller values of ρ, we recommend to precompute and remember the set of possible subpaths for a given local configuration so as to speed up the uniform picking of the reconfiguration. As subpaths are clamped at both ends, this should not impact too much the memory. Note however that if a free path (whose only one end is clamped) belongs to the hexagon, it might however require too much computation time as there might be too many possible configurations to consider (as it could be as long as $\Theta(\rho^2)$ if it fills a constant fraction of the hexagon). In this situation, we recommend to allow only the δ last beads of the free path to be rerouted.

4 Conclusion

In this article, we have proposed a new model which includes randomly occurring local reconfigurations in the oritatami model. As demonstrated by the experimental implementation of the tile assembly model [12, 14], taking into account this natural phenomenon into model is a necessity to design successful in-vitro implementations. We hope to implement this model as an open source software soon, and are eager to explore whether the basic structures that made computation possible in the oritatami model, such as glider, switchback, folding meter and pocket, can be made tolerant to such thermal noise. One may also wonder if such reconfiguration could be exploited to discover new way of computing using co-transcriptional molecular folding.

References

1. C.G. Evans, *Crystals That Count! Physical Principles and Experimental Investigations of DNS Tile Self-Assembly*. Ph.D. thesis (California Institute of Technology, 2014)
2. C. Geary, G. Grossi, E.K.S. McRae, P.W.K. Rothemund, E.S. Andersen, RNA origami design tools enable cotranscriptional folding of kilobase-sized nanoscaffolds. Nat. Chem. **13**, 549–558 (2021)
3. C. Geary, P.E. Meunier, N. Schabanel, S. Seki, Oritatami: a computational model for molecular co-transcriptional folding. Int. J. Mole. Sci. **20**(9), 2259 (2016) (Its conference version was published in Proc. MFCS 2016)
4. C. Geary, P.-E. Meunier, N. Schabanel, S. Seki, Proving the Turing universality of oritatami cotranscriptional folding. in *Proceedings of ISAAC 2018, LIPIcs*, vol. 123, (2018), pp. 23:1–23:13
5. C. Geary, P.W.K. Rothemund, E.S. Andersen, A single-stranded architecture for cotranscriptional folding of RNA nanostructures. Science **345**, 799–804 (2014)
6. K. Maruyama, S. Seki, Counting infinitely by oritatami cotranscriptional folding. Nat. Comput. **20**(2), 329–340 (2021)
7. E.C. Merkhofer, P. Hu, T.L. Johnson, Introduction to cotranscriptional RNA splicing. Methods Mole. Biol. **1126**, 83–96 (2014)
8. R. Parales, D. Bentley, "co-transcriptionality"—the transcription elongation complex as a nexus for nuclear transactions. Mole. Cell **36**(2), 178–191 (2009)
9. D. Pchelina, N. Schabanel, S. Seki, Y. Ubukata, Simple intrinsic simulation of cellular automata in oritatami molecular folding model, in *Proceedings of LATIN2020, LNCS*, vol. 12118 (Springer, 2020), pp. 425–436
10. N. Schabanel, Simple OS simulator. http://perso.ens-lyon.fr/nicolas.schabanel/OSsimulator/
11. K.E. Watters, E.J. Strobel, A.M. Yu, J.T. Lis, J.B. Lucks, Cotranscriptional folding of a riboswitch at nucleotide resolution. Nat. Struct. Mole. Biol. **23**(12), 1124–1131 (2016)
12. E. Winfree, R. Bekbolatov, Proofreading tile sets: error correction for algorithmic self-assembly, in *DNA Computing, 9th International Workshop on DNA Based Computers, DNA9, Madison, WI, USA, June 1–3, 2003*, revised Papers, eds. by J. Chen, J.H. Reif. Lecture Notes in Computer Science, vol. 2943 (Springer, 2003), pp. 126–144. https://doi.org/10.1007/978-3-540-24628-2_13

13. D. Woods, H.L. Chen, S. Goodfriend, N. Dabby, E. Winfree, P. Yin, Active self-assembly of algorithmic shapes and patterns in polylogarithmic time, in *Proceedings of ITCS2013* (ACM, 2013), pp. 353–354
14. D. Woods, D. Doty, C. Myhrvold, J. Hui, F. Zhou, P. Yin, E. Winfree, Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. Nature **567**, 366–372 (2019)