Oritatami systems
doodle uncomputably

Daria Pchelina (LIPN)
Nicolas Schabanel (LIP)

Shinnosuke Seki (UEC, Tokyo)
Guillaume Theyssier (I2M)
Oritatami: A model for co-transcriptional folding

The program:
- a sequence of bead types (the transcript)

The instructions:
- the rule \( a \heartsuit b \) if bead types \( a \) and \( b \) attract each other

The input configuration:
- Some beads placed beforehand (the seed)
RNA Folding
(Real time: ~1 second)

Part been folded

Part already folded

Encoding of the transcript

Video: Geary
Oritatami: A model for co-transcriptional folding

The dynamics

• Starting from the seed, the sequence is produced one bead at a time

• Only the δ last produced beads are free to move and explore the accessible positions to settle in the ones maximizing the number of bonds

• All other beads remain in their last locations

here, delay δ = 3

Geary, Meunier, Schabanel, Seki MFCS 2016
Oritatami: A model for co-transcriptional folding

The dynamics

• Starting from the seed, the sequence is produced one bead at a time

• Only the δ last produced beads are free to move and explore the accessible positions to settle in the ones maximizing the number of bonds

• All other beads remain in their last locations

here, delay δ = 3

Geary, Meunier, Schabanel, Seki MFCS 2016
Oritatami: A model for co-transcriptional folding

The dynamics.

• Starting from the seed, the sequence is produced one bead at a time

• Only the \( \delta \) last produced beads are free to move and explore the accessible positions to settle in the ones maximizing the number of bonds

• All other beads remain in their last locations
Oritatami: A model for co-transcriptional folding

The dynamics.

- Starting from the seed, the sequence is *produced one bead at a time*
- Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones *maximizing the number of bonds*
- All other beads remain in their last locations
**Oritatami:**
A model for co-transcriptional folding

The dynamics.

- Starting from the seed, the sequence is *produced one bead at a time*.
- Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones *maximizing the number of bonds*.
- All other beads remain in their last locations.

*Geary, Meunier, Schabanel, Seki MFCS 2016*
Oritatami: A model for co-transcriptional folding

The dynamics.

- Starting from the seed, the sequence is *produced one bead at a time*.
- Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones *maximizing the number of bonds*.
- All other beads remain in their last locations.
Oritatami: A model for co-transcriptional folding

The dynamics.

- Starting from the seed, the sequence is *produced one bead at a time*.
- Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones *maximizing the number of bonds*.
- All other beads remain in their last locations.

Geary, Meunier, Schabanel, Seki MFCS 2016
Oritatami: A model for co-transcriptional folding

The dynamics.

- Starting from the seed, the sequence is *produced one bead at a time*.
- Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones *maximizing the number of bonds*.
- All other beads remain in their last locations.

Configuration(s) with max. bonding
Oritatami: A model for co-transcriptional folding

The dynamics.

• Starting from the seed, the sequence is produced one bead at a time

• Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones maximizing the number of bonds

• All other beads remain in their last locations
Oritatami: A model for co-transcriptional folding

The dynamics.

• Starting from the seed, the sequence is *produced one bead at a time*

• **Only the $\delta$ last produced beads** are free to move and explore the accessible positions to settle in the ones **maximizing the number of bonds**

• All other beads remain in their last locations

There might be **several configurations** with max. bonding

Geary, Meunier, Schabanel, Seki MFCS 2016
Oritatami: 
A model for co-transcriptional folding

The dynamics.

- Starting from the seed, the sequence is produced one bead at a time

- Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones maximizing the number of bonds

- All other beads remain in their last locations

The bead has same position in all maximal extension $\Rightarrow$ deterministic

There might be several configurations with max. bonding
Oritatami: A model for co-transcriptional folding

The dynamics.

• Starting from the seed, the sequence is produced one bead at a time

• Only the $\delta$ last produced beads are free to move and explore the accessible positions to settle in the ones maximizing the number of bonds

• All other beads remain in their last locations

The bead has same position in all maximal extension $\Rightarrow$ deterministic

There might be several configurations with max. bonding

Geary, Meunier, Schabanel, Seki MFCS 2016
Previous work

Some abstract Tile Assembly seminal work

- Tile assembly systems are Turing universal [Winfree, 1998]
- Arbitrary shape assembly with optimal tile set size [Soloveichik, Winfree, 2007]
- Intrinsic universality [Doty et al, 2012]
- Uncomputable limit configuration [Lathrop et al, 2011]

Oritatami

- A binary counter [Geary, Meunier, S., Seki, 2016]
- Heighdragon fractal [Masuda, Seki, Ubukata, 2018]
- Folding arbitrary shapes [Demaine et al, 2018]
- NP-hardness for oritatami design [Geary et al, 2016; Ota, Seki, 2017; Han, Kim, 2017] and for non-deterministic oritatami equivalence [Han et al, 2016]
- Efficient Turing Machine simulation through tag-systems [Geary et al, 2018]
- Intrinsic 1D Cellular Automata simulation [Pchelina et al, 2020]
Previous work

Some abstract Tile Assembly seminal work

• Tile assembly systems are Turing universal [Winfree, 1998]
• Arbitrary shape assembly with optimal tile set size [Soloveichik, Winfree, 2007]
• Intrinsic universality [Doty et al, 2012]
★ Uncomputable limit configuration [Lathrop et al, 2011]

Oritatami

• A binary counter [Geary, Meunier, S., Seki, 2016]
• Heighdragon fractal [Masuda, Seki, Ubukata, 2018]
• Folding arbitrary shapes [Demaine et al, 2018]
• NP-hardness for oritatami design [Geary et al, 2016; Ota, Seki, 2017; Han, Kim, 2017] and for non-determinisitic oritatami equivalence [Han et al, 2016]
• Efficient Turing Machine simulation through tag-systems [Geary et al, 2018]
• Intrinsic 1D Cellular Automata simulation [Pchelina et al, 2020]
★ TODAY: Uncomputable limit configuration & Turmite intrinsic simulation
Uncomputable?  
Limit configuration?

- The limit configuration $c^\infty$ is the configuration at the end of time:

$$c^0 \subset \cdots \subset c^t \subset c^{t+1} \subset \cdots \subset c^\infty = \bigcup_{t=0}^{\infty} c^t$$

- $c^\infty$ is uncomputable is the function $(i, j) \mapsto c_{i,j}^\infty$ is uncomputable
Uncomputable limit configuration in aTAM

- Simulate in parallel all Turing machines on an empty input
- Go down to place a (black) tile at the bottom if the TM halts

⇒ The bottom row is uncomputable
Uncomputable limit configuration in aTAM

- Simulate in parallel all Turing machines on an empty input
- Go down to place a (black) tile at the bottom if the TM halts

⇒ The bottom row is uncomputable
Uncomputable limit configuration in aTAM

- Simulate in parallel all Turing machines on an empty input
- Go down to place a (black) tile at the bottom if the TM halts

⇒ The bottom row is uncomputable
Uncomputable limit
Oritatami configuration

- Introducing Turmites
- Turmites doodle uncomputably
- Delay-3 Oritatami systems simulate Turmites intrinsically
Turmites

A finite automata follows a self-avoiding path, moving and writing a state according to a uniform local rule.

A clockwise walker

The rule:
A finite automata follows a self-avoiding path, moving and writing a state according to a uniform local rule.

A clockwise walker

The rule:
Turmites implement CA

Left/Right Swiping

The rule:
The rule:

Left/Right Swiping

Turmites implement CA
Turmites doodle uncomputably
Turmites doodle uncomputably

Diagonal simulation of all Turing Machines
Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$

Diagonal simulation of all Turing Machines

$T_{M_2}$ halts
Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$

Diagonal simulation of all Turing Machines

$TM_2$ halts
Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$

Diagonal simulation of all Turing Machines

$\text{TM}_2$ halts
Markers are regularly placed of increasing size \( \approx i + i(i + 1)/2 \)

Diagonal simulation of all Turing Machines

\( TM_2 \) halts

Turmites doodle uncomputably
Turmites doodle uncomputably

Markers are regularly placed of increasing size \( \approx i + i(i+1)/2 \)

Diagonal simulation of all Turing Machines

\( \text{TM}_2 \) halts
Markers are regularly placed of increasing size \( \approx i + \frac{i(i + 1)}{2} \).

Diagonal simulation of all Turing Machines.
Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$

Diagonal simulation of all Turing Machines

$\text{TMs}_2$ halts
Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx i + \frac{i(i + 1)}{2}$

Diagonal simulation of all Turing Machines

$\text{TM}_2$ halts
Markers are regularly placed of increasing size $\approx i + i(i+1)/2$

Diagonal simulation of all Turing Machines

$\Phi$
Turmites doodle uncomputably

Markers are regularly placed of increasing size \( \approx i + i(i + 1)/2 \)

Diagonal simulation of all Turing Machines

\( \text{TM}_3 \) halts

\( \text{TM}_2 \) halts
Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx \frac{i + i(i + 1)}{2}$

Diagonal simulation of all Turing Machines

$TM_3$ halts

$TM_2$ halts
Turmites doodle uncomputably

Markers are regularly placed of increasing size \( \approx i + i(i + 1)/2 \)

Diagonal simulation of all Turing Machines

\( TM_3 \) halts \( TM_2 \) halts
Markers are regularly placed of increasing size $\approx i + \frac{i(i+1)}{2}$
Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$

Diagonal simulation of all Turing Machines
Diagonal simulation of all Turing Machines

Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$
Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx i + \frac{i(i+1)}{2}$

Diagonal simulation of all Turing Machines

$\text{TM}_1$ halts $\quad \text{TM}_3$ halts $\quad \text{TM}_2$ halts
Théorème. Turmites doodle uncomputably

Markers are regularly placed of increasing size $\approx i + i(i + 1)/2$

Diagonal simulation of all Turing Machines

$TM_1$ halts $TM_3$ halts $TM_2$ halts
Oritatami systems simulate Turmites intrinsically
Oritatami systems simulate 1D CA intrinsically.

- **Previous work.** [PSSU, 2020]
  1D Cellular automata intrinsic simulation

Diagram:
1. **Init** $4Qx$
2. **Scaffold** $4Qx'$
3. **Read** $x,y$
4. **Lookup table** shifted by $\Delta_{xy} = 2(Qx + y)$
5. Absorb the offset $2(Qx + y)$
6. Special beads in the lookup table flip matching magnets to write $x',y'$

Table:
- Time: Down
- Flat Shift: Down
Oritatami systems simulate
1D CA intrinsically

- **Previous work.** [PSSU, 2020]
  1D Cellular automata intrinsic simulation

2 Problems. Supercells must be **isotropic**
We need to **exit from an arbitrary side**...
the Supercell
the Supercell
the Supercell

1. Scaffold layer
the Supercell

1. Scaffold layer
2. Read layer
3. Write layer
the Supercell

1. Scaffold layer
2. Read layer
3. Write layer
4. Exit layer
the Supercell

1. Scaffold layer
2. Read layer
If $j$–th bit $= 1$, then Offset $+= 2^jQ^i$

$\Rightarrow$ Offset on $i$–th side $= \text{state}(i) \times Q^i$

$\Rightarrow$ Total Offset on all side $= \langle \text{states} \rangle \in 0..Q^6–1$
the Supercell

1. Scaffold layer
2. Read layer
3. Write layer
Writing. Offset pulls the transition table to the right

- bit 0 and bit 1 fold differently

- The boxes hide the $Q^6$ unused entries in the transition table
Writing. Offset pulls the transition table to the right

bit 0 and bit 1 fold differently
⇒ the exit layer shows or hide the special beads

The boxes hide the $Q^6$ unused entries in the transition table
the Supercell

1. Scaffold layer
2. Read layer
3. Write layer
the Supercell

1. Scaffold layer
2. Read layer
3. Write layer

Resynchronize
Can absorb any offset $\leq Q^6$

*(in Zig-Zags! 😓)*
the Supercell

1. Scaffold layer
2. Read layer
3. Write layer
4. Exit layer
Exiting… or not

By default, *exit layer* follows the border of the exit box
Exiting… or not

With the proper signal (offset!), exit layer folds upon itself and… exit!
Exiting… or not

With the proper signal (offset!), exit layer folds upon itself and… exit!
Key new tool: Folding meter

it folds upon itself into boxes
Suspiciously simple fact

All layers stay synchronized
Almost there
Almost there
Conclusion

- Furthermore, Oritatami & Turmite doodles can have any density expressible in... \( \Pi_2 \) (?)
- No need for parallelism
- No need for 3D
- Lines just don't cross!
- Some more work and we'll have an running implementation!
- New tools for Oritatami: **Folding meter, oubliettes, distant sensor & crazy speedbumps!**