Published in final edited form as:

New Gener Comput. 2015 July ; 33(3): 271–295. doi:10.1007/s00354-015-0302-7.

Dynamic Simulation of 1D Cellular Automata in the Active aTAM

Nataša Jonoska,

Department of Mathematics and Statistics, University of South Florida, Tampa, FL, 33620 USA

Daria Karpenko, and

Department of Mathematics and Statistics, University of South Florida, Tampa, FL, 33620 USA

Shinnosuke Seki^{*1}

Helsinki Institute for Information Technology (HIIT), Department of Computer Science, Aalto University, P. O. Box 15400, FI-00076, Aalto, FINLAND

Nataša Jonoska: jonoska@mail.usf.edu; Daria Karpenko: dkarpenko@mail.usf.edu; Shinnosuke Seki: s.seki@uec.ac.jp

Abstract

The Active aTAM is a tile based model for self-assembly where tiles are able to transfer signals and change identities according to the signals received. We extend Active aTAM to include deactivation signals and thereby allow detachment of tiles. We show that the model allows a dynamic simulation of cellular automata with assemblies that do not record the entire computational history but only the current updates of the states, and thus provide a way for (a) algorithmic dynamical structural changes in the assembly and (b) reusable space in self-assembly. The simulation is such that at a given location the sequence of tiles that attach and detach corresponds precisely to the sequence of states the synchronous cellular automaton generates at that location.

Keywords

DNA Self-Assembly; Activa aTAM; Cellular Automaton; Signaling; Dynamic Simulation; Synchronization

1 Introduction

Algorithmic DNA self-assembly has been demonstrated to assemble a variety of nanoscale structures representing computations by a range of devices including a binary counter,²⁾ logic circuits,¹¹⁾ transducers,³⁾ and simulation of cellular automata (CA) for fractal patterns.⁵⁾ In almost all cases, the assembled structure contains not just the tiles corresponding to the current computation of the system but also the whole history of the computation starting with the input. Such non-reusable space in the computation simulation is a result of the DNA building blocks designed to simulate computation. The blocks (or in the most common theoretical model, aTAM,¹⁶⁾ tiles) contain single stranded sticky ends that attach to their complementary counter parts on other blocks (tiles) and build up the structure

^{*1}Current address: Department of Communication Engineering and Informatics, University of Electro-Communications, 1-5-1, Chofugaoka, Chofu, Tokyo, 1828585, JAPAN.

one step at a time. Thus the structure grows while recording each step of the computation. The algorithmic assembly, rather than a random assembly, is obtained by introduction of a *temperature* parameter, a non-negative integer indicating the necessary number of sticky ends for an assembly event. At temperature values higher than one, this parameter enforces cooperative block binding events.

Recent experiments demonstrate that the assembly sequence can be controlled with protective strands attached to the sticky ends such that the sticky ends become active only when an appropriate strand (called a “signal”) displaces the protective strand through a strand displacement mechanism.^{6, 14, 17)} This has resulted in the expansion of the Winfree's theoretical model aTAM to the Active aTAM⁹⁾ and signal passing TAM (STAM).¹³⁾ Active aTAM was motivated by the idea of signals being transmitted across a building block similarly as in the use of strand displacement mechanism for the “walkers,”⁶⁾ while STAM was mainly motivated by the idea that activation of a signal at one side of the block initiates the activation of a signal on another side of the block.¹⁴⁾ With both of these models, it has been shown that universal computation can be achieved at temperature one,^{9, 13)} which has been suspected to be unachievable in standard aTAM.¹²⁾ The model STAM is more general than Active aTAM in the sense that it allows for signals to deliver not just activation, but also deactivation of the sticky ends. This deactivation mechanism in STAM was used to describe a strict assembly of the Sierpinski triangle¹³⁾ and in shape replication mechanisms.¹⁰⁾

The first aim of this paper is to expand the Active aTAM to incorporate glue deactivation, which allows assemblies to become unstable after signal passing events, and, as a result, fall apart. This allows Active aTAM systems to compute and to dynamically change the assembly configuration during the computation process, without storing and recording the entire history of the computation. We demonstrate this feature by showing how Active aTAM with deactivation can simulate a one-dimensional cellular automaton (CA). CAs have been an archetype for investigating self-assembly since the introduction of the aTAM when Winfree simulated a 1D blocked CA in aTAM.¹⁶⁾ In a stronger sense of simulation that is similar to intrinsic universality, Hendricks and Patitz designed an aTAM simulator for 2D nondeterministic CAs.⁸⁾ Due to the lack of a tile detachment mechanism in aTAM, these simulators increase assembled structures with each time step as the computation proceeds. For example, in aTAM, a simulation of the first t steps of a 1D CA starting from an input of length m needs an array of size $O((m + 2t)t)$. This is necessary because in order to simulate the synchronous update of CA by chemical self-assembly which is an inherently asynchronous process, these simulators need to have tiles that assemble cooperatively. The cooperative assembly is obtained by the positive integer temperature parameter of aTAM, and it is assumed possible if and only if the temperature is at least 2.¹²⁾ In contrast, arrays of only width 3 and length $(m + 2t + 4)$ are needed for the Active aTAM 1D CA simulator proposed in this paper. Moreover, the simulation is at temperature 1 because the signal design eliminates the necessity for tile cooperation. Furthermore, the proposed system allows not only space but also detached tiles to be reused, since after detachment, addition of fuel strands,^{15, 17)} or other external control,⁷⁾ can return the tiles to precisely the same state as they were before their involvement in the computation. Though the simulation we present in this paper is for deterministic CAs, it can be modified for nondeterministic ones, as discussed briefly in the concluding section (Section 4).

The reusable space in our simulator is similar to that of the STAM simulation of a Turing machine (TM) by Padilla et al.,¹³⁾ which consumes 7 tiles for each step of the TM computation. The simulation presented here assumes a reuse of the tiles and there is presumably no tile consumption. Furthermore, the computational models being simulated are different. For one step of a CA computation, all cells of the CA must be updated in a synchronized manner, while it suffices to update a single cell per computational step of a TM. Our simulator updates as many tiles as there are non-quiescent cells of the current CA configuration, and, in fact, does not need more tiles than that. In comparison with other CA simulations, our simulation is dynamic, the tiles do not necessarily remain within the structure during the assembly but tiles with different identities appear at the same location of the assembly. The sequence of tile identities at a given location follows precisely the trace of states of the corresponding cellular automaton cell, hence although inherently asynchronous, the self-assembly process simulates the state changes of the synchronously updated cellular automaton cells.

This paper is organized as follows. In Section 2, we modify the original definition of Active aTAM to incorporate glue deactivation. The section also contains a schematic for proposed experimental attachment/detachment implementation. In its subsection, we also compare the modified Active aTAM with STAM. Section 3 is devoted to our Active aTAM simulator for deterministic 1D CAs. In Section 3.1, we verify the correctness of the simulation (Propositions 3.1 and 3.2). Section 4 concludes the paper by summarizing the results and discussing further research directions.

2 Active aTAM with Deactivation

We refer the reader to ⁹⁾ for a detailed description of the Active aTAM and to 1) for general background on algorithmic self-assembly. Here we extend the Active aTAM to accommodate deactivation. The elements of a finite set Σ are called *glue labels*, or just *glues*. We define $\Sigma^- = \{-\sigma \mid \sigma \in \Sigma\}$. Refer to Fig. 1.

An *active tile* is a triple $\tau = (t, \mathcal{A}, \mathcal{S})$ where t is a tile without signaling: a unit square whose sides are denoted by $t_{+y}, t_{+x}, t_{-y}, t_{-x}$ indicating their orientations on the coordinate plane.^{*2} We set $\mathcal{D} = \{+y, +x, -x, -y\}$ to be the set of directions. Each side of the tile is of the form (A, I) , consisting of sets of *active* and *inactive* glue labels, where $A, I \subset \Sigma \cup \Sigma^-$, a finite set of labels and complements marked with + and -, respectively. Each label has a corresponding strength via the function $s : \Sigma \rightarrow \mathbb{Z}^+$. In this paper, we assume a constant function for s such that $s(\Sigma) = \{1\}$. The tile τ is equipped with a *activation/deactivation* signal pathway set $\mathcal{A} \subset \hat{\Sigma}$ and a *transmission* signal pathway set $\mathcal{S} \subset \hat{\Sigma}$, where

$\hat{\Sigma} = \{(c, \text{state})_i^j \mid c \in \Sigma, j \in \mathcal{D}, i \in \mathcal{D} \cup \{0\}, \text{state} = 0, 1\}$ is a set of all signals. The signal $(c, \text{state})_i^j$ indicates that a signal for label c is transmitted from direction i to direction j , while $i = 0$ denotes the initiation of a signal for label c in direction j . Signals with state 1 are activation signals and signals with state 0 are deactivation signals.

^{*2}We drift from the standard E, N, W, S notation to ease the exposition.

For a set of active tiles $\mathcal{T}$, an *assembly map* $\alpha : \mathbb{Z}^2 \rightarrow \mathcal{T}$ is a partial map with a finite connected domain. A *bond* is a set of glue labels, and we say that there is a bond between tiles τ and τ' in direction $d \in \mathcal{D}$ that contains $\sigma \in \Sigma$ if τ and τ' appear adjacent in α in direction d and σ is active at side t_d in τ and $-\sigma$ is active at t_{-d} in τ' . The strength of a bond is the sum of the strengths of all glues contained in that bond. An assembly map is θ -stable if it cannot be broken into two or more assembly maps without breaking bonds whose sum of strengths is at least θ . A *tile assembly instance* for a specified temperature parameter $\theta \in \mathbb{Z}$ is a θ -stable assembly map. A *tile assembly* is an equivalence class of tile assembly instances under translation and a *tile assembly representative* is an instance selected to represent the tile assembly. A *unit assembly instance* is an assembly instance with a singleton domain. We also abuse the notation and we refer to a unit assembly as a *tile*.

Define the *sum of two assembly maps* α and α' as the map $\beta = \alpha \oplus \alpha'$ if β is an assembly map with $\text{dom } \beta = \text{dom } \alpha \cup \text{dom } \alpha'$ and $\text{dom } \alpha \cap \text{dom } \alpha' = \emptyset$. We say that β is *partitioned* to α and α' . If $\beta = \alpha \oplus \alpha'$ is θ -stable, and α' is a tile, we say that β is obtained from α by a tile attachment. We also say that β *contains* τ , (resp. α) if $\beta_z = \tau$ for some $z \in \mathbb{Z}^2$ (resp. $\beta|_P = \alpha$ for some $P \subset \text{dom } \beta$).

For an assembly map α , consider the set of all partitions of α into the set of tiles that together form α . Call a partition $\mathcal{P}(\alpha)$ θ -stable if for every set in $\mathcal{P}(\alpha)$, the sum of the tiles in that set is θ -stable. Define a partial order on the θ -stable partitions of α as follows: $\mathcal{P}_1(\alpha) \leq \mathcal{P}_2(\alpha)$ if $|\mathcal{P}_1(\alpha)| \leq |\mathcal{P}_2(\alpha)|$ and every set in $\mathcal{P}_2(\alpha)$ is a subset of a set in $\mathcal{P}_1(\alpha)$. Under this partial order, the set of all θ -stable partitions of α contains a unique maximal element, which we call the *maximal stable partition of α* and denote with $\mathfrak{Q}_\alpha$. The uniqueness follows from the fact that if α, α' are stable and $\text{dom } \alpha \cap \text{dom } \alpha' = \emptyset$, then $\alpha|_P \oplus \alpha'|_{P'}$ is stable, where $P = \text{dom } \alpha \setminus (\text{dom } \alpha \cap \text{dom } \alpha')$ and respectively $P' = \text{dom } \alpha' \setminus (\text{dom } \alpha \cap \text{dom } \alpha')$. Intuitively, $\mathfrak{Q}_\alpha$ is the set of largest stable components forming α . In particular, when α is an assembly instance, we have $\mathfrak{Q}_\alpha = \{\alpha\}$.

A *tile modification function* (TMF) is defined to act on individual tiles in a tile assembly instance, changing their signal and label sets depending on their immediate neighbors. Inactive labels are moved to the active set and active labels are moved to the inactive set, as appropriate, while transmission signals are added or removed as they are propagated.

A tile modification function f is defined as follows. Let α be a tile assembly instance and denote $\alpha(z)$ by α_z . For $z \in \mathbb{Z}^2$ and $\alpha_z = \tau = (t, \mathcal{A}, \mathcal{S})$, let $\tau_{+y} = \alpha_{z+(0,1)}$, $\tau_{+x} = \alpha_{z+(1,0)}$, $\tau_{-y} = \alpha_{z+(0,-1)}$, $\tau_{-x} = \alpha_{z+(-1,0)}$ define the neighbors of τ on the respective sides. If τ_d is not defined for some $d \in \mathcal{D}$ write $\tau_d = \varepsilon$, otherwise write $\tau_d = (t^d, \mathcal{A}^d, \mathcal{S}^d)$.

Let $N = \{(0, 0), (0, 1), (0, -1), (1, 0), (-1, 0)\}$. We define a function $f: \mathcal{T}^5 \rightarrow \mathcal{T}$ called a *tile modification function* $f(\alpha_z + N) = f(\tau, \tau_{+y}, \tau_{+x}, \tau_{-y}, \tau_{-x}) = \alpha'_z = (t', \mathcal{A}', \mathcal{S}')$ that transforms the active tile $\tau = \alpha_z$ into another active tile α'_z with the following rules, one modifying the active glues on τ 's sides and the other modifying the set of transmission signals. Let $t = (t_{+y}, t_{+x}, t_{-y}, t_{-x})$ and $t' = (t'_{+y}, t'_{+x}, t'_{-y}, t'_{-x})$.

Before specifying the details, we illustrate the activation signal transmission with the following example.

Example 2.1—Refer to Fig. 2 (a). The t_{+x} side of the active tile shown contains one active label from the Σ^- set (a black square marked with “-”) and two inactive labels, one from Σ^+ , p (a white square marked with “+ p ”) and another from Σ^- , $-q$ (a white square marked with “- q ”). The side t_{+y} contains two active labels from Σ^+ and no inactive labels. The side t_{-x} contains no active or inactive labels. The side t_{-y} contains one active label from Σ^- . The activation signal pathway set contains two signal pathways: p_{-x}^{+x} and q_{-y}^{+x} . The transmission signal pathway set contains three signal pathways: a_0^{-x} , b_{-y}^{-x} , and c_{+x}^{+x} .

In the same figure (b)-(d), an example of activation signal transmission is depicted. Suppose two tiles attach as shown in Fig. 2 (b). In order to apply the tile modification function f to the tile τ_1 , we consider its $+x$ neighbor τ_2 . The tile τ_2 has a transmission pathway a_{-x}^{+x} and an initiation signal b_0^{-x} . This means that on τ_2 , the initiation signal a_0^{+x} can be added (since it will be transmitted from τ_1) and to the b_{+x}^{+x} transmission pathway an initiation signal b_0^{+x} is added on τ_1 . Both new initiation signals are depicted in (c). Now, we apply f to the assembly in (c). The b_0^{+x} initiation signal on τ_1 activates the previously inactive $+x$ glue label $-b$ on τ_2 using its b_{-x}^{+x} activation signal pathway. The resulting completed assembly is shown in (d).

Tile Side Modification: Let $i \in \mathcal{D}$ and $t_i = (A, I)$. Then we define

$t'_i = ((A \setminus C) \cup B, (I \setminus B) \cup C)$, where B consists of inactive labels that become active, and C is the set of active labels that become inactive. Specifically,

$$B = \left\{ c \in I \mid \exists (|c|, 1)_d^i \in \mathcal{A}, (|c|, 1)_0^{-d} \in \mathcal{S}^d \text{ for some } d \in \mathcal{D} \right\} \text{ and } \nexists d' \in \mathcal{D} \text{ s.t. } \exists (|c|, 0)_{d'}^i \in \mathcal{A}, (|c|, 0)_0^{-d'} \in \mathcal{S}^{d'} \Big\},$$

$$C = \left\{ c \in A \mid \exists (|c|, 0)_d^i \in \mathcal{A}, (|c|, 0)_0^{-d} \in \mathcal{S}^d \text{ for some } d \in \mathcal{D} \right\} \text{ and } \nexists d' \in \mathcal{D} \text{ s.t. } \exists (|c|, 1)_{d'}^i \in \mathcal{A}, (|c|, 1)_0^{-d'} \in \mathcal{S}^{d'} \Big\}.$$

Informally, if the tile side contains an inactive glue labeled c with a corresponding activation signal pathway, and an adjacent tile contains a corresponding initiation signal, then the glue labeled c becomes active (first line of B), unless there is a deactivation signal being sent for the same label at the same time, in which case the state of the label does not change (second line of B). If the tile side contains an active glue labeled c with a corresponding deactivation signal pathway, and an adjacent tile contains a corresponding deactivation initiation signal, then the glue labeled c becomes inactive (first line of C), unless there is an activation signal from a neighboring tile being sent for the same label at the same time, in which case the state of the label does not change (second line of C). Thus, glues are activated or deactivated when the appropriate activation or deactivation signal is present on a neighboring tile. In the case that both a deactivation signal and an activation signal for the same glue are present simultaneously on one or more of the neighboring tiles, no action is taken by the TMF.

Transmission Signal Modification: For $i, j \in \mathcal{D}$ and state = 0, 1, the new set of transmission signals in t' becomes $\mathcal{S}' = (\mathcal{S} \cup \mathcal{S}_{\text{added}}) \setminus \mathcal{S}_{\text{removed}}$, where

$$\mathcal{S}_{\text{added}} = \left\{ (c, \text{state})_0^j \mid \exists d \in \mathcal{D} \text{ with } (c, \text{state})_d^j \in \mathcal{S} \text{ and } (c, \text{state})_0^{-d} \in \mathcal{S}^d \right\},$$

$$\mathcal{S}_{\text{removed}} = \left\{ (c, \text{state})_0^j \mid \exists i \in \mathcal{D} \text{ such that } (c, \text{state})_i^j \in \mathcal{S} \text{ but } \nexists d \in \mathcal{D} \text{ with } (c, \text{state})_d^j \in \mathcal{S} \text{ and } (c, \text{state})_0^{-d} \in \mathcal{S}^d \right\}.$$

Informally, a tile receiving a signal is equivalent to it acquiring a corresponding initiation signal via $\mathcal{S}_{\text{added}}$. In the case of tile t being a part of a subassembly of a larger assembly that broke apart, an initiation chain could be broken. A broken chain is detected if a tile has both an initiation and a transmission signal of the same kind but none of its neighbors have a corresponding initiation signal; in that case, the “broken” initiation signal is removed (see right-hand assembly in Fig. 3 (d), (e)).

An *active tile assembly system* (Active TAS) is an ordered triple $(\mathcal{T}, s_0, \theta)$ where $\mathcal{T}$ is a finite set of tiles, s_0 is the seed tile assembly, and $\theta \in \mathbb{Z}^+$ is the temperature parameter of the system.

We extend f to an assembly map $\alpha : \mathbb{Z}^2 \rightarrow \mathcal{T}$ in a specified Active TAS by applying f simultaneously to a subset $P \subset \text{dom } \alpha$ of active tiles in α to obtain a maximal θ -stable partition: $f(\alpha) = \mathcal{Q}_\alpha$ where $\alpha'_z = f(\alpha_{z+N})$ for all $z \in P$. If α is an assembly instance, i.e., θ -stable, and $f(\alpha) = \{\alpha\}$ for every $P \subset \text{dom } \alpha$, then we say that α is a *completed assembly instance*. We say that a completed assembly instance α is a *completion* of the assembly map β if there is a sequence $\beta = \alpha^0, \alpha^1, \dots, \alpha^k = \alpha$ such that for each $i \geq 1$, $\alpha^{i+1} \in f(\alpha^i)$. At each step, there may be many stable assemblies produced, but we consider only one maximal θ -stable assembly, and discard all others.

Given an Active TAS $(\mathcal{T}, s_0, \theta)$, a *producible assembly sequence* is a sequence $\bar{\alpha} = (\alpha^0, \alpha^1, \dots)$ where $\alpha^0 = s_0$ is the seed tile positioned at $(0, 0)$ and for every $i \geq 1$, α^{i+1} is either $f^k(\alpha^i)$ for some $k > 0$ containing the seed tile s_0 , or (α^{i+1}) is an assembly map obtained from α^i by attachment of a single tile. The definition for producible assembly sequences allows new tiles to bind to “incomplete” assemblies. However, we note that producible assembly sequences also contain $\bar{\alpha} = (\alpha^0, \alpha^1, \dots)$ where α^{i+1} is a completion of $(\alpha^i)'$ when it is chosen that f is applied on the whole domain of α^i and k completes the signal transmission.

Example 2.2—For detachment signals consider the example in Fig. 3 and its caption. The figure also shows a producible assembly sequence. Suppose α^0 , the seed, consists of τ_1 in (a), placed at $(0, 0)$, and α^1 is its sum with τ_2 in (a), then α^2 resulting from the addition of τ_3 in (a). Two activation signals labeled a and b are transmitted from τ_1 to τ_2 and one deactivation signal c is transmitted from τ_2 to τ_1 . The resulting α_3 assembly is shown in (b) where appropriate initiation signals have been added to the tiles. In (c), the deactivation signal c on τ_2 deactivates the glue between τ_2 and τ_3 , which yields a non stable assembly map. This non stable map consists of two stable components $\tau_1 \tau_2$ and τ_3 depicted in (d). Since τ_1 is the seed, the next assembly map α_4 taken in the producible assembly sequence is the complex $\tau_1 \tau_2$. In (e), the signal transmission can continue on both assembly maps, a

break for the signal b is detected and tile τ_3 has the initiation signal for b removed. However, there are no signal transmissions on α_4 and the assembly $\tau_1 \tau_2$ remains unchanged.

2.1 Signal Transmission in STAM and Active aTAM

A possible signal transmission scheme is shown in Fig. 4, where in (i) an attachment scheme is presented and (ii) depicts a detachment scheme. In (i), attached tile A contains a strand with a toehold α' and a sequence complementary to α on the green strand attached to tile B . Through branch migration as originally demonstrated by Yurke, et al.,¹⁷⁾ a sticky end on the other side of tile B becomes available and allows attachment of a third tile C . This scheme was recently experimentally demonstrated in Padilla, et al.¹⁴⁾ where a signal was transmitted along five tiles controlling the cascading assembly event. In (ii), a similar detachment scheme is depicted. An attachment of tile C to tile B detaches tile A on the other side of tile B . The sticky end of A remains the same as before its attachment to tile B . Applying additional signal strands using toehold x' can release the sticky end xa on tile B and allow next attachment of A .

Both Active aTAM⁹⁾ and STAM¹³⁾ can be seen as extensions of the conventional aTAM being equipped with a signaling mechanism to control the state of glue labels. Originally these models were defined as an extension of 2HAM (two handed assembly model),⁴⁾ where the assembly growth does not need a seed and it can be achieved by attachments of larger structures, not just a single tile. The assembly restriction to single tile addition at one step define those models as extensions of aTAM. The two models, Active aTAM and STAM, differ in the following aspects.

The signal propagation protocol of the two models is different. In Active aTAM, an initiation signal is transferred by establishing a directed path of transmission signal pathways of the corresponding kind. Tiles along this path are being equipped with the initiation signal one after another, and in the end, the initiation signal thus transferred affects the target glue. What if the path is cut before the target receives the signal, for example, by detachment? On the resulting segment between the cutting point and the target end, initiation signals start vanishing in order, one after another, from the cut to the target end. Thus, unless further tiles attach at the target end to relay the dying signal, or the path is refilled with initiation signals by attaching another source to the cutting point, the signal is lost without affecting its target. In contrast, in STAM, a signal on a tile can affect a label only on the same tile. A signal is fired when a tile binds, and it is pending on that tile until being 'processed'; when it is 'processed,' it changes the state of its target label *on the same tile* accordingly. In order to transmit a signal at a distance, a signal is fired at the initial tile and processed to activate one of the tile's labels in order for the tile to bind to the next tile, and this binding event fires a signal from the second tile allowing it to bind to the third one, and proceeds in this way until the signal reaches the target. Thus, even if the initial tile is detached before the target is affected, the attachment of the other tiles allows for the signal to reach its target. Given this difference in the handling of source loss, it is unlikely that the dynamics of the signal transmission of an Active aTAM system could be simulated by the dynamics of the signal transmission of STAM.

Another difference between the models concerns the signal update. In Active aTAM, signals on many tiles of an assembly, in fact on any subset of the assembly domain, are updated simultaneously. In contrast, in STAM, a tile is chosen at random and a signal pended on it is chosen randomly and processed; so, at most one tile is updated (i.e., one of its glues changes its state) at a time step (instead of a tile being updated, other actions possible for a time step include combining two assemblies into one θ -stable assembly or disassembling an assembly into a θ -stable subassemblies). It is not clear, however, whether this particular difference in dynamics constitutes a difference in the computational power of the models.

In contrast to the original definition of STAM, we have defined the glue deactivation in the Active aTAM so that deactivated glues can be activated again. One can envision designing the sticky ends of the tiles with a scheme similar to the “seesaw” gates.¹⁵⁾ The glue reactivation may require additional energy used by the system. This energy can be generated by applying fuel strands similarly to the Yurke model,¹⁷⁾ or fluorescent labeling on some of the strands and laser illumination.⁷⁾ The definition of STAM can be similarly modified, although in its original definition, deactivated labels remain inactive for the remainder of the assembly process. Glue reactivation allows for not only the reuse of space but also the reuse of tiles. An experimental verification showing the feasibility of this mechanism could be a significant advancement for the field.

3 An Active aTAS that Dynamically Simulates a 1D CA

Let $\mathbb{A}$ be a 1-dimensional deterministic cellular automaton with $\Sigma = \{0, 1, \dots, n-1\}$ its state set, a local function $\ell: \Sigma^3 \rightarrow \Sigma$ and $G_\ell: \Sigma^{\mathbb{Z}} \rightarrow \Sigma^{\mathbb{Z}}$. Let $\bar{x}$ represent an initial configuration for $\mathbb{A}$ with a finite number of non-zero states, $\bar{x} = (x_0, \dots, x_m)$, such that its left- and right-most nonzero states are contained in $\bar{x}$.

The tile set $\mathcal{T} = \mathcal{T}(\mathbb{A}, \bar{x})$ consists of four general types of tiles: frame tiles (Fig. 5), input tiles (Fig. 6), top computational tiles (Fig. 7), and bottom computational tiles (Fig. 8). The frame tiles contain the seed tile and serve solely to hold the input and computational tiles in a single 1-stable assembly instance as well as transfer signals between different time step computations on both sides of the frame. The input tiles are used to set up a given input and to initiate the computation. The computational tiles alternate between the two sides of the frame which we call top and bottom. Both top and bottom computational tiles further alternate between tiles of two levels, 0 and 2 at the bottom, and 1 and 3 at the top. Although level 0 and level 2 computational tiles can appear next to each other at the bottom of a single assembly, their cross communication is prevented because no signal can cross between different levels. Similarly with the top tiles of levels 1 and 3. Bottom level 0 (resp. 2) tiles send activation signals for level 1 at the top (resp. 3) across the frame and deactivate glues for level 3 (resp. 1). Similarly, top level 1 (resp. 3) tiles send activation signals across the frame for level 2 at the bottom (resp. 0) and deactivate labels for level 0 (resp. 3). These signal transmissions across the frame tiles are summarized in Fig. 9. The signals at the levels are such that neither activation nor deactivation signals can cross the frame at a given position z unless there are three tiles (at positions $z-1$, z and $z+1$) present above or below of the same level. Those signals check the identity of the three neighboring tiles and activate

a glue that corresponds to the value of ℓ across the frame. Each input and computational tile has a unique state of $\mathbb{A}$ assigned to it via a correspondence function $g = g_{\ell}$

$$g: \mathcal{T} \rightarrow \Sigma$$

using ℓ and each tile's parameters. An experimental scheme showing attachment of a level 1 tile and detachment of a level 0 tile is depicted in Fig. 10. We formally define the function g and tile types below. Note that in our description, we make use of two indices: $z_2 \in \mathbb{Z}_2$ as the parity index and $z_4 \in \mathbb{Z}_4$ as the level index such that adding integers to z_2 and z_4 is done mod 2 and mod 4, respectively. The parity index discriminates between adjacent tiles, marking them as odd or even, and allows the tiles to distinguish a signal coming from its immediate neighbor as opposed to a signal from one tile over. The level index discriminates among computation tiles corresponding to different time steps, modulo 4.

The general idea of the construction is as follows: a single line of frame tiles assembles with $m + 1$ (input length) unique *input frame tiles* and left and right extension *general frame tiles* with an alternating z_2 index. The exposed active glues on the bottom of the input frame tiles correspond to the sequence of *input tiles*; the exposed active glues on the bottom of general frame tiles correspond to state 0 input tiles. The input tiles are set up to “filter” signals coming from other input tiles that are their immediate neighbors and send them through the frame to activate the glue (at the top of the frame) corresponding to the computed tile: a level 1 computational tile. Once three neighboring level 1 computational tiles are attached at the top of the frame, a deactivation signal for the input tiles is sent through and able to deactivate and detach the input tile that is no longer needed (all possible input tile deactivation signals are sent simultaneously; the level 1 tiles do not need to identify which tiles were used in their computation). At the same time, all possible activation signals for the next time step are filtered through the level 1 tiles and the unique glue corresponding to the computed level 2 computational tile is activated on the bottom of the frame in place of the deactivated input tile glue. The neighboring level 2 tiles send deactivation signals for level 1 and activation signals for level 3 to the top of the frame (in each case, the signals have to pass through 3 neighboring tiles before they can pass through, thus ensuring that deactivation takes place only when the previous level tile is no longer needed in any computation). Level 3 tiles send deactivation signals for level 2 and activation signals for level 0 to the bottom of the frame; and level 0 tiles send deactivation signals for level 3 and activation signals for level 1 to the top of the frame.

Note that this construction allows input or level 0 computational tiles to be adjacent to level 2 tiles at the bottom of the frame and level 1 to level 3 at the top of the frame. This does not cause any problem since the signals are separated by level, e.g., level 2 deactivation signals can only travel through level 3 tiles. Moreover, adjacent tiles are at most 2 time steps apart, e.g., two level 3 tiles cannot be next to each other unless they correspond to the same time step; thus, there is no signal mixing among time steps, ensuring the correctness of the computation. More detail on this is given in the proofs following the formal description of the tiles.

Frame tiles—Refer to Fig 5. For each parity index $z_2 \in \mathbb{Z}_2$, define a general frame tile, $\tau_{\text{frame}, z_2} = ((t_{+y}, t_{+x}, t_{-y}, t_{-x}), \mathcal{A}, \emptyset)$, which functions to extend the frame to the left and serves to increase the computational spaces if necessary by allowing input tiles corresponding to state 0 of $\mathbb{A}$ to attach to its $-y$ side.

$$\begin{aligned} t_{+x} &= (\{(F, z_2)\}, \emptyset) \\ t_{-x} &= (\{-(F, z_2+1)\}, \emptyset) \\ t_{+y} &= (\emptyset, \{(i, j, k, z_2, 2l+1) | i, j, k \in \Sigma, l=0, 1\}) \\ t_{-y} &= (\{-(0, z_2)\}, \{(i, j, k, z_2, 2l) | i, j, k \in \Sigma, l=0, 1\}). \end{aligned}$$

The set of (de)activation signals $\mathcal{A}$ consists of all possible signals that cross between the computational tiles on both sides of the frame:

$$\begin{aligned} \mathcal{A} &= \{(i, j, k, z_2, 2l+1), m\}_{-y}^{+y} | i, j, k \in \Sigma, l, m \\ &= 0, 1\} \cup \{(i, j, k, z_2, 2l), m\}_{+y}^{-y} | i, j, k \in \Sigma, l, m=0, 1\} \cup \{(0, z_2), 0\}_{+y}^{-y} \} \end{aligned}$$

The index z_2 allows the $z_2 = 0$ (even) frame tiles to attach to $z_2 = 1$ (odd) frame tiles at the $t_{\pm x}$ sides. Observe that the inactive glue labels on $t_{\pm y}$ sides are 5-tuples consisting of all possible triples of states drawn from Σ , the parity index z_2 , and the level index z_4 (even levels 0, 2 on the bottom, odd levels 1, 3 at the top). These glues correspond to computational tiles (defined later) each of which has a unique active complementary glue. The active glue on the t_{-y} side corresponds to the input “state 0” tile with the matching parity index.

Define an analogous right general frame tile, τ_{frame, z_2} , for each $z_2 = 0, 1$, by replacing the labels (F, z_2) and $-(F, z_2 + 1)$ with $(F, z_2)'$ and $-(F, z_2 + 1)'$ respectively above. These tiles serve to extend the frame to the right. In addition to the four general frame tiles, for input $\bar{x} = (x_0, \dots, x_m)$, we define the following $m + 1$ “input” frame tiles, $\tau_{x_i, \text{frame}}$ that specify the glues for the input tile sequence at the bottom, as follows.

The tile $\tau_{x_0, \text{frame}}$ is designated as the seed tile s_0 for the system and has the same t_{+y} and t_{-x} sides as the $z_2 = 0$ general left frame tile, as well as the same signal sets except that the deactivation signal corresponding to $(0, z_2)$ is replaced with one corresponding to $(x_0, 0)$. The side $t_{+x} = (\{F_i\}, \emptyset)$. The side t_{-y} has the same inactive label set as in the general frame tile, but a different active one: $A = \{-(x_0, 0)\}$.

For $i = 1, \dots, m - 1$, the tile $\tau_{x_i, \text{frame}}$ has the same t_{+y} side and signal sets as the $z_2 = i \pmod{2}$ general frame tile with a deactivation signal replacement of $(0, z_2)$ with $(x_i, i \pmod{2})$. The side $t_{+x} = (\{F_i\}, \emptyset)$ and $t_{-x} = (\{-F_{i-1}\}, \emptyset)$. The side t_{-y} has $A = \{-(x_i, i \pmod{2})\}$ and the same inactive label set as the general frame tile.

Lastly, $\tau_{x_m, \text{frame}}$ has the same t_{+y} and t_{+x} sides as the $z_2 = m \pmod{2}$ general right frame tile and the same signal sets with a deactivation signal replacement of $(0, z_2)$ with $(x_m, m \pmod{2})$.

2)). The side $t_{-x} = (\{-F_{m-1}\}, \emptyset)$. The side t_{-y} has the same inactive label set as in the general frame tile and $A = \{-(x_m, m \pmod{2})\}$.

The glues on the $t_{\pm x}$ sides of the input frame tiles are designed such that the frame assembles in the particular sequence from the seed and allows general frame tiles to attach at either end of the sequence. The inactive glues and signals function exactly as in the general frame tiles, but the active glues are made to uniquely match the input tiles defined below.

Input Tiles—Refer to Fig 6. For each x_i in $\bar{x}$, the input sequence tile τ_{x_i} consists of a single active label $(x_i, i \pmod{2})$ on the t_{+y} side and the following transmission signal set, all being activation signals:

$$\begin{aligned} \mathcal{S} = & \left\{ ((j, k, x_i, i+1 \pmod{2}), 1, 1)_0^{-x} | j, k \in \Sigma \right\} [\text{initiate identity to the left}] \\ & \cup \left\{ ((j, x_i, k, i \pmod{2}), 1, 1)_{+x}^{-x} | j, k \in \Sigma \right\} [\text{transmit identity to the left}] \\ & \cup \left\{ ((x_i, j, k, i+1 \pmod{2}), 1, 1)_{+x}^{+x} | j, k \in \Sigma \right\} [\text{transmit identity right}] \\ & \cup \left\{ ((j, x_i, k, i \pmod{2}), 1, 1)_{-x}^{+y} | j, k \in \Sigma \right\} [\text{transmit identity up}] \end{aligned}$$

This set creates a signal filtering system for adjacent tiles. Each tile initiates a set of signals testing for all possible pairs of neighboring tile states (j, k) with its state identity x_i fixed in the third coordinate, the opposite parity indicated in the fourth coordinate, and the next level of computation $(\pmod{4})$ in the fifth coordinate (in this case, 1). At the same time, the tile transmits to the left all signals from its right neighbor that match its state in the second coordinate and its parity (and its right neighbor's state in the third coordinate). The incoming signals from the left that do not match its parity must be signals initiated two tiles over to the right. The tile chooses the signal that matches its state in the first coordinate to redirect back to its immediate neighbor on the right. These signals received from the left are transmitted by the tiles to the frame and activate a unique corresponding glue on a frame tile's top.

We also include two “input 0” tiles, $\tau_{(0, z_2)}$ for $z_2 \in \mathbb{Z}_2$ defined exactly as the input tiles but replacing x_i with 0 and i with z_2 . These ensure that the input sequence can be extended arbitrarily far to the left and right with 0 states on the frame for the purposes of computing arbitrarily large time steps. The correspondence function g for all input tiles is defined as $g(\tau_{x_i}) = x_i$ and

$$g(\tau_{(0, z_2)}) = 0.$$

Top Computational Tiles—Refer to Fig. 7. For each fixed triple of states $i_0, j_0, k_0 \in \Sigma$, $z_2 \in \mathbb{Z}_2$, and $z_4 \in \{1, 3\}$, let $\ell_0 = \ell(i_0, j_0, k_0)$ and define the top computational tile $\tau_{i_0, j_0, k_0, z_2, z_4}$ - the tile corresponding to the state ℓ_0 , i.e., the state computed from the triple of states (i_0, j_0, k_0) - as the tile with the active glue $-(i_0, j_0, k_0, z_2, z_4)$ on the t_{-y} side and the following set of transmission signals (with z_2 and z_4 addition defined modulo 2 and 4 respectively). We set $\mathcal{S} = \mathcal{S}_{\text{activate}} \cup \mathcal{S}_{\text{deactivate}}$ with

$$\begin{aligned}\mathcal{S}_{\text{activate}} &= \left\{ ((j, k, \ell_0, z_2+1, z_4+1), 1)_0^{-x} | j, k \in \Sigma \right\} \cup \left\{ ((j, \ell_0, k, z_2, z_4+1), 1)_{+x}^{-x} | j, k \in \Sigma \right\} \cup \left\{ ((\ell_0, j, k, z_2+1, z_4+1), 1)_{+x}^{+x} | j, k \in \Sigma \right\} \\ \mathcal{S}_{\text{deactivate}} &= \left\{ ((i, j, k, z_2+1, z_4-1), 0)_0^{-x} | i, j, k \in \Sigma \right\} \cup \left\{ ((i, j, k, z_2, z_4-1), 0)_{+x}^{-x} | i, j, k \in \Sigma \right\} \cup \left\{ ((i, j, k, z_2+1, z_4-1), 0)_{+x}^{+x} | i, j, k \in \Sigma \right\}\end{aligned}$$

Since the initial tiles have different glues on the $+y$ side than the computational tiles, for $z_4 = 1$ add the signals $((i, z_2+1), 0)_0^{-x}$ and $((i, z_2+1), 0)_{+x}^{-x}$ for each label (i, z_2+1) that appears on an input tile and signals $((i, z_2), 0)_{+x}^{-x}$ and $((i, z_2), 0)_{-x}^{-y}$ for each label (i, z_2) that appears on an input tile. The main signaling set creates a filtering system for activation signals similar to that of the input tiles. Also, each tile sends deactivation signals for all possible previous level tiles to the $-y$ side. The filtering ensures that all three top tiles that were computed using a tile from the bottom row are in place before that bottom tile can detach.

Bottom Computational Tiles—Refer to Fig. 8. For each fixed triple of states $i_0, j_0, k_0 \in \Sigma$, $z_2 \in \mathbb{Z}_2$, and $z_4 \in \{0, 2\}$, the bottom computational tile $\tau_{i_0, j_0, k_0, z_2, z_4}$ is defined as the tile with the active glue $(i_0, j_0, k_0, z_2, z_4)$ on the t_{+y} side and a transmission signal set $\mathcal{S}$ defined exactly as for the top tiles but with a change in the orientation on the (de)activation signals

$\begin{matrix} -y & +y \\ -x & -x \end{matrix}$

defined from $-x$ to $-x$.

The correspondence function g for the top and bottom computational tiles is defined as

$$g(\tau_{i, j, k, z_2, z_4}) = l(i, j, k).$$

Take $\mathcal{T}$ to be the tile set consisting of the above tiles. Let $(\mathcal{T}, s_0, 1)$ be the TAS corresponding to $(\mathbb{A}, \bar{x})$.

3.1 Verification of the Simulation

Let $G_t^t(\bar{x})_i$ represent the state of cell i at time step t when $\mathbb{A}$ acts on $\bar{x} = (x_0, \dots, x_m)$, where $G_t^0(\bar{x})_i = x_i$ for $i = 0, \dots, m$ and $G_t^0(\bar{x})_i = 0$ otherwise.

Proposition 3.1—Let $t \in \mathbb{Z}^+$ and consider $G_t^t(\bar{x})$. There exists a producible assembly sequence $\bar{\alpha} = (\alpha^0, \dots, \alpha^k)$, $\alpha^0 = s_0 = \tau_{x_0, \text{frame}}$, with a subsequence $\alpha^{j_0}, \dots, \alpha^{j_t}$ such that for $z \in \mathbb{Z} \setminus [-t, m+t]$, $\alpha^{j_i}(z, \pm 1)$ is empty and for $z \in [0, m]$ and $0 \leq i \leq t$,

$$G_t^i(\bar{x})_z = \begin{cases} g(\alpha^{j_i}(z, -1)) & \text{if } i \text{ is even} \\ g(\alpha^{j_i}(z, 1)) & \text{if } i \text{ is odd.} \end{cases}$$

Proof—Fix $t \in \mathbb{Z}^+$. We provide an explicit construction for the claimed assembly sequence and subsequence. This assembly sequence is a sequence of completion assemblies. Refer to Section 2 for the definition of a sum of two assembly maps. Let $\alpha^0 = s_0 = ((0, 0) \mapsto \tau_{x_0, \text{frame}})$. For $i = 1, \dots, m$, let α^i be the completion of $\alpha^{i-1} \oplus ((i, 0) \mapsto \tau_{x_i, \text{frame}})$, which is

possible since the input frame tiles have matching glues on the $\pm x$ sides. Therefore, $\alpha^0, \dots, \alpha^m$ is a producible assembly sequence and α^m is defined by $\alpha^m(i, 0) = \tau_{x_i, \text{frame}}$.

Now, for $i = 1, \dots, t$, define α^{m+i} to be the completion of the assembly map $\alpha^{m+i-1} \oplus ((m+i, 0) \mapsto \tau_{\text{frame}, m+i \pmod{2}})$ and α^{m+t+i} to be the completion of $\alpha^{m+t+i-1} \oplus ((-i, 0) \mapsto \tau_{\text{frame}, i \pmod{2}})$. Again, by construction of the frame tiles, $\alpha^0, \dots, \alpha^{m+2t}$ is a producible assembly sequence (refer to Fig. 11) and α^{m+2t} is the input frame sequence of m tiles with t general frame tiles on either side.

From here, the input tiles can be attached one by one. Define α^{m+2t+i} for $i = 1, \dots, m+2t$ as the completion of

$$\alpha^{m+2t+i-1} \oplus \begin{cases} (z, -1) \mapsto \tau_{xz} & \text{if } \alpha^{m+2t+i-1}(z, 0) = \tau_{x_i, \text{frame}} \\ (z, -1) \mapsto \tau_{0, z \pmod{2}} & \text{otherwise} \end{cases}$$

where $z = -t - 1 + i$. This, and the following is illustrated in Fig. 12. When $i = 3$ and τ^{i-2} , τ^{i-1} , and τ^i are the bottom tiles added in the $m+2t+i-2$, $m+2t+i-1$, and $m+2t+i$ assembly sequence steps respectively, the completion of the assembly map sum in step $m+2t+i$ yields the activation of the glue labelled $((g(\tau^{i-2}), g(\tau^{i-1}), g(\tau^i), -t-1+i-1 \pmod{2}), 1), 1)$ in the frame tile at coordinate $(-t-1+i-1, 0)$. Recall that this is precisely the glue corresponding to the top tile τ such that $g(\tau) = \ell(g(\tau^{i-2}), g(\tau^{i-1}), g(\tau^i)) = G_t^1(\bar{x})_{-t-1+i-1}$ therefore the completion of the assembly at step $\alpha^{m+2t+m+2t}$ yields a row of $m+2t$ input tiles attached to the row of $m+2t$ frame tiles with $m+2t-2$ glues active at the top of the frame tiles corresponding to the next level (level 1) top tiles that represent precisely the states computed by $\mathbb{A}$ from the input states. So, we take $\alpha^{m+2t+m+2t}$ to be α^0 the first assembly in the subsequence (Fig. 13).

The next assembly in the subsequence, α^1 , can be constructed analogously by attaching the $m+2t-2$ top level 1 computational tiles one at a time from the left. By the same argument as before, the attachment of three consecutive tiles in locations $(i-2, 1)$, $(i-1, 1)$, $(i, 1)$ activates the glue corresponding to the level 2 bottom computational tile τ for location $(i-1, -1)$ such that $g(\tau) = G_t^2(\bar{x})_{i-1}$ and deactivates the glue for the input tile at $(i-1, -1)$ (by sending all possible deactivation signals for the input tiles of the matching parity). The assembly α^1 thus consists of the $m+2t$ frame tiles, $m+2t-2$ top computational level 1 tiles, and two input level bottom tiles on either edge of the frame (these cannot detach since they do not have three level 1 top tiles directly above). See Fig. 14.

To construct α^2 , the $m+2t-4$ level 2 tiles are attached one at a time from the left. Since level 2 signals cannot be sent through input level tiles, three consecutive level 2 tiles are required before any activation or deactivation takes place. Thus, α^2 consists of the $m+2t$ frame tiles, with two input bottom tiles on either side of $m+2t-4$ level 2 bottom tiles, and two level 1 top tiles on either side of a gap of length $m+2t-6$ for level 3 top tiles to come. See Fig. 15. By construction, the active level 3 glue in location $(i, 1)$ corresponds to the tile τ such that $g(\tau) = G_t^3(\bar{x})_i$. Note that the level 3 top tiles send signals to activate level 0 bottom

tiles which correspond to $t = 4$. Continuing the construction analogously for t steps, the conclusion follows inductively (Fig. 16).

For the converse, we consider the *trace* of the CA $\mathbb{A}$ at location z , $\{G_l^t(\bar{x})_z\}_{t=0}^\infty$. We show that for every producible assembly, the sequence of tiles that appear at the z th location corresponds precisely to the trace of $\mathbb{A}$ at z .

For each tile assembly a that assembles in the TAS $\mathcal{T}(\mathbb{A}, \bar{x})$, let its representative be the instance α such that $\alpha(0, 0) = s_0$. Let $\bar{\alpha} = (\alpha^0, \alpha^1, \dots)$ be a producible assembly sequence (of representatives) such that for some i , α^i has a nonempty position $(z, 1)$ or $(z, -1)$. Note that for all i , $\text{dom } \alpha^i \subset \mathbb{Z} \times \{-1, 0, 1\}$.

For fixed z and $\bar{\alpha}$, define sequence of tiles indexed with even numbers, $(\alpha^{j_{2k}}(z, -1))_{k=0}^\infty$ where $j_0 = \min\{i \mid \alpha^i(z, -1) \text{ is nonempty}\}$ and $j_{2(k+1)} = \min\{i > j_{2k} \mid \alpha^i(z, -1) \text{ is nonempty and there is } i' > j_{2k}, \alpha^{i'}(z, -1) \text{ is empty}\}$. In other words, this is a sequence of all bottom row tiles that appear in a fixed location z units away from the seed in attachment events (separated by detachment events).

Define also an analogous sequence of top row tiles with odd indexes $(\alpha^{j_{2k+1}}(z, 1))_{k=0}^\infty$ where $j_1 = \min\{i \mid \alpha^i(z, 1) \text{ is nonempty}\}$ and $j_{2(k+1)+1} = \min\{i > j_{2k+1} \mid \alpha^i(z, 1) \text{ is nonempty and there is } i' > j_{2k+1}, \alpha^{i'}(z, 1) \text{ is empty}\}$.

Proposition 3.2—For the even and odd sequences defined above, for all $k = 0, 1, \dots$,

$$g(\alpha^{j_{2k}}(z, -1)) = G_l^{2k}(\bar{x})_z \quad \text{and} \quad g(\alpha^{j_{2k+1}}(z, 1)) = G_l^{2k+1}(\bar{x})_z$$

where g is the correspondence function. Moreover, for all k it holds

$$j_{2k} < j_{2k+1} < j_{2(k+1)}.$$

That is, the result of every $k + 1$ 'th attachment of a nonframe tile in location z corresponds exactly to the k 'th time step state in location z of $\mathbb{A}$.

Proof—Fix $z \in \mathbb{Z}$. It suffices to consider all possible transitions from α^i to α^{i+1} in an assembly sequence $\bar{\alpha} = (\alpha^0, \alpha^1, \dots)$. Note that if α^{i+1} is a sum of α^i and a frame tile, then $\alpha^{i+1}(z, \pm 1) = \alpha^i(z, \pm 1)$. By construction, the location $(z, 1)$ can only be affected by tiles at $(z, 1)$, $(z + 1, 1)$, $(z + 2, 1)$, $(z - 1, 1)$, $(z, -1)$, and $(z \pm 1, -1)$ because signals initiated by any computational tile can travel at most 2 units horizontally to the left, 1 to the right, and to travel vertically have to pass through a frame tile. A frame tile at $(z, 0)$ can only transmit signals to $(z, 1)$ that have traveled through tiles at $(z, -1)$ and $(z \pm 1, -1)$; the claim follows. The case for $(z, -1)$ is symmetric. So, if α^{i+1} follows from α^i by attachment of a computational tile outside this neighborhood, then $\alpha^{i+1}(z, \pm 1) = \alpha^i(z, \pm 1)$.

Consider the location $(z, -1)$. If $\alpha^i(z, -1) = \varepsilon$ and $\alpha^{i+1}(z, -1) = \tau$ such that $i + 1$ is the least index in the assembly sequence $\bar{a}$ with $(z, -1)$ nonempty, then τ is an input tile. Setting $\alpha^0 = \alpha_{i+1}$ yields by definition $g(\alpha^{j_0}(z, -1)) = G_{\ell}^0(\bar{x})_z$. Suppose a tile at $(z, -1)$ has level index $z_4 \in \mathbb{Z}_4$. Note that it can only be detached if three next-level index $z_4 + 1$ tiles are attached at $(z, -1, 1)$, $(z, 1)$, and $(z + 1, 1)$, which themselves require the tile in question to be present for the attachment to occur. It follows that any two consecutive attachment events at location $(z, 1)$ in an assembly sequence are separated by an attachment event at $(z, -1)$. Since the situation is symmetric when considering a tile at $(z, 1)$, it follows that a strict progression of levels is enforced for a given tile location (coordinate) in an assembly sequence. Moreover, by the signal filtering mechanism described earlier, the “next level” $z_4 + 1$ tile is the one computed from the three same level z_4 neighbors using the rules in $\mathcal{A}$ defined by ℓ . Thus, if $g(\alpha^{j_{2k}}(z, -1)) = G_{\ell}^{2k}(\bar{x})_z$ then $g(\alpha^{j_{2k+1}}(z, 1)) = G_{\ell}^{2k+1}(\bar{x})_z$ and $g(\alpha^{j_{2k+2}}(z, -1)) = G_{\ell}^{2k+2}(\bar{x})_z$. The conclusion follows.

3.2 Complexity Analysis of the Simulation

Tile complexity—For an n state automaton with an input of length $m + 1$, the simulator consists of the following tile types:

- 4 general frame tiles $\tau_{\text{frame},0}, \tau_{\text{frame},1}, \tau_{\text{rframe},0}, \tau_{\text{rframe},1}$;
- $m + 1$ input-specific frame tiles $\tau_{x_0,\text{frame}}, \tau_{x_1,\text{frame}}, \dots, \tau_{x_m,\text{frame}}$;
- $m + 3$ input tiles $\tau_{x_0}, \dots, \tau_{x_m}, \tau_{(0,0)}, \tau_{(0,1)}$;
- $4n^3$ top computational tiles $\tau_{i_0,j_0,k_0,z_2,z_4}$ with $i_0, j_0, k_0 \in \Sigma$, $z_2 \in \mathbb{Z}_2$, and $z_4 \in \{1, 3\}$;
- $4n^3$ bottom computational tiles $\tau_{i_0,j_0,k_0,z_2,z_4}$ with $i_0, j_0, k_0 \in \Sigma$, $z_2 \in \mathbb{Z}_2$, and $z_4 \in \{0, 2\}$.

Thus, its tile complexity is $8n^3 + 2m + 8$.

Label complexity—Frame tiles have 1 label on the $t_{\pm x}$ sides, $2n^3$ labels on t_{+y} side, and $2n^3 + 1$ labels on t_{-y} side. Input tiles and bottom computational tiles have only one label on the t_{+y} side. Top computational tiles also contain only one label rather on the t_{-y} side. In summary, it is the t_{-y} side of the frame tiles that is provided with the largest number ($2n^3 + 1$) of labels.

Signal pathway complexity—As shown in Table 1, computational tiles require largest number ($4n^3 + 4n^2 + 4n$) of pathways among all tiles for the simulation. For the elementary CA over alphabet $\{0, 1\}$, this number is 56.

Space complexity—The current simulation can extend its frame leftward and rightward arbitrarily long within any limited number of steps t . Thus, we cannot argue the space complexity of this simulation.

We can modify the simulation so as to use only $3(m + 5 + 2t)$ positions in order to simulate the first t steps, as long as we are concerned with only initial configurations $\bar{x} = (x_0, x_1, \dots,$

x_m) all but whose cells $x_0, x_1, \dots, x_m$ are quiescent, which is a state q such that $\ell(q, q, q) = q$. Starting with such a configuration, at the first step, at most two cells to the left of the cell x_0 can transit to a non-quiescent state, and similarly, at most two cells to the right of x_m . Generally speaking, $t + 1$ cells to the left of x_0 as well as $t + 1$ cells to the right of x_m are necessary and sufficient in order to simulate the first t steps. We first introduce four frame tiles $\tau_{x-2, \text{frame}}, \tau_{x-1, \text{frame}}, \tau_{x_{m+1}, \text{frame}}, \tau_{x_{m+2}, \text{frame}}$, which assemble the following initial frame:

$$\tau_{x-2, \text{frame}} - \tau_{x-1, \text{frame}} - \tau_{x_0, \text{frame}} - \tau_{x_1, \text{frame}} - \dots - \tau_{x_m, \text{frame}} - \tau_{x_{m+1}, \text{frame}} - \tau_{x_{m+2}, \text{frame}}.$$

The labels at the both ends of this frame, that is, $-(F, 1)$ at the t_x side of $\tau_{x-2, \text{frame}}$ and $(F, m + 2 \pmod{2})$ at the t_x side of $\tau_{x_{m+2}, \text{frame}}$, are set inactive so that the initial frame cannot extend leftwards or rightwards until these glues are activated. See Fig. 17. We modify input tiles so as to fire a signal to activate the $-(F, 1)$ label and to have it propagate in parallel to the signal for state transition. That is to say, when the three input tiles compute the next state of the position x_{-1} and propagate the appropriate signal for that, the modified input tiles also fire a signal and propagate it as shown in Fig. 17 in order to activate the label $-(F, 1)$ so that at time step 1, a left general frame tile can attach to the left of $\tau_{x-2, \text{frame}}$.

Left general frame tiles ($z_2 = 0, 1$) must be modified by setting its label $-(F, z_2 + 1)$ at the t_x side be inactive initially and by being equipped with pathways $-(F, 1), 1)_{+x}^{-x}$ and $-(F, 0), 1)_{+y}^{-x}$ if $z_2 = 0$, or $-(F, 0), 1)_{+x}^{-x}$ and $-(F, 1), 1)_{-y}^{-x}$ otherwise. These pathways propagate an activation signal launched by input or computational tiles up to the label at the t_x side for further leftward frame extension. The signal to activate $-(F, 1)$ is launched by input or bottom computation tiles as exemplified above, while the signal to activate $-(F, 0)$ is launched by top computation tiles in an analogous manner.

The right general frame tiles are to be modified analogously so as to extend the frame rightwards by one position at a step of CA to be simulated.

This modification introduces the four frame tiles, and hence, increases the tile complexity by four. It does not increase the label complexity. It adds two pathways to frame tiles, four pathways to input tiles and to top/bottom computational tiles.

4 Conclusion

The generalization of Active aTAM to incorporate glue deactivation seems natural and with a possible experimental scheme. We used this feature to simulate 1D deterministic CAs in a dynamic way by reusing tiles and space. Its advantage over the existing CA simulators is the dynamic update of tile assemblies according to the respective CA state updates. The system does not need to build-up on the recorded space-time computation (extended 2d array) in order to identify next tile assembly as a result of the CA computation. This concept of the simulation is similar to the Padilla et al.'s simulation of TMs¹³⁾ in terms of reusing space. However, in the dynamic CA simulation, we have shown a way to preserve the sequence of

synchronously updated cells in the CA through an inherently asynchronous computational model such as the Active aTAM.

The system presented here can be easily made nondeterministic, adjusted to simulate nondeterministic CAs. As discussed in Section 3, from the triple (i_0, j_0, k_0) encoded in the bottom (resp. top) glue, each top (resp. bottom) computational tile determines its next state to be $\ell_0 = \ell(i_0, j_0, k_0)$ (state transition). If the CA to be simulated is nondeterministic and, for example, admits two possible next states ℓ_0, ℓ'_0 from the triple, we introduce two computational tiles corresponding to the triple such that the first one transits to ℓ_0 and the other to ℓ'_0 .

There are at least two future research directions. Theoretically, it may be of interest to identify an Active aTAM system that dynamically simulates 2-dimensional CAs. It also remains open to obtain experimental verification for the combination of glue deactivation and reactivation, as has been done for activation individually.¹⁴⁾

Acknowledgments

This work has been supported in part by NSF grants CCF-1117254 and DMS-0900671, and the NIH grant R01GM109459-01 to N. J. and by Academy of Finland, Postdoctoral Researcher Grant No. 13266670/T30606 to S. S.

References

1. Self-assembly wiki. <http://self-assembly.net>
2. Barish RD, Rothmund PWK, Winfree E. Two computational primitives for algorithmic self-assembly: Copying and counting. *Nano Lett.* 2005; 5(12):2586–2592. [PubMed: 16351220]
3. Chakraborty B, Jonoska N, Seeman NC. A programmable transducer self-assembled from DNA. *Chem Sci.* 2012; 3(1):168–176. [PubMed: 23139854]
4. Cheng Q, Aggarwal G, Goldwasser MH, Kao MY, Schweller RT, de Espans PM. Complexities for generalized models of self-assembly. *SIAM Journal on Computing.* 2005; 34:1493–1515.
5. Fujibayashi K, Hariadi R, Park S, Winfree E, Murata S. Toward reliable algorithmic self-assembly of DNA tiles: A fixed-width cellular automaton pattern. *Nano Lett.* 2008; 8(7):1791–1797. [PubMed: 18162000]
6. Gu H, Chao J, Xiao SJ, Seeman NC. A proximity-based programmable dna nanoscale assembly line. *Nature.* 2010; 465(7295):202–205. [PubMed: 20463734]
7. He X, Sha R, Mi Y, Chaikin P, Seeman N. Self-replication, exponential growth, selection and competition in systems of DNA origami tiles. *Science.* 2015 under review.
8. Hendricks J, Patitz M. On the equivalence of cellular automata and the tile assembly model. *Proc MCU 2013, EPTCS.* 2013; 128:167–189.
9. Jonoska N, Karpenko D. Active tile self-assembly, part 1: Universality at temperature 1. *Int J Found Comput Sci.* 2014; 25:141–163.
10. Keenan, A.; Schweller, R.; Zhong, X. *Proc DNA19 - 2013, LNCS.* Vol. 8141. Springer; 2013. Exponential replication of patterns in the signal tile assembly model; p. 118–132.
11. Mao C, LaBean TH, Reif JH, Seeman NC. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. *Nature.* 2000; 407:493–496. [PubMed: 11028996]
12. Meunier PE. A pumping lemma for non-cooperative self-assembly. *arXiv:* 1312.6668. 2014
13. Padilla, JE.; Patitz, MJ.; Pena, R.; Schweller, RT.; Seeman, NC.; Sheline, R.; Summers, SM.; Zhong, X. *Proc UCNC 2013, LNCS.* Vol. 7956. Springer; 2013. Asynchronous signal passing for tile self-assembly: Fuel efficient computation and efficient assembly of shapes; p. 174–185.

14. Padilla JE, Sha R, Kristiansen M, Chen J, Jonoska N, Seeman NC. A signal-passing DNA-strand-exchange mechanism for active self-assembly of DNA nanostructures. *Angewandte Chemie International Edition*. 2015; doi: 10.1002/anie.201500252
15. Qian L, Winfree E. A simple dna gate motif for synthesizing large-scale circuits. *Journal of The Royal Society Interface*. 2011
16. Winfree, E. Ph D thesis. California Institute of Technology; Jun. 1998 Algorithmic Self-Assembly of DNA.
17. Yurke B, Turberfield AJ, Mills AP, Simmel FC, Neumann JL. A DNA-fuelled molecular machine made of DNA. *Nature*. 2000; 406(6796):605–608. [PubMed: 10949296]

Biographies

Nataša Jonoska: She is a professor at the Department of Mathematics and Statistics at University of South Florida in Tampa Florida, USA. Her research interests are in theoretical and computational models of molecular self-assembly and molecular biology. She has been awarded the Rozenberg Tulip Award in DNA Computing and Molecular Programming by the International Society for Nanoscale Science and Computing in 2007, and elected as AAAS Fellow in 2014. Currently, she serves as the Chair of the Steering Committee for the annual DNA Computing and Molecular Programming conference and also co-chairs the steering committee of the annual Unconventional Computing and Natural Computing conference. She serves on editorial board of several journals including *Theoretical Computer Science*, *Natural Computing*, *Computability*, *International Journal of Foundations of Computer Science*, and has edited several books on these topics.

Daria Karpenko, Ph.D.: She attended the University of South Florida from 2007 to 2015, completing B.A. programs in Philosophy and Physics and B.A./M.A. and Ph.D. programs in Mathematics. Her research interests include algorithmic self-assembly, biological computing, and the mathematical theory of computation.

Shinnosuke Seki, Ph.D.: He is an assistant professor at the Department of Communication Engineering and Informatics at University of Electro-Communications in Chofu, Tokyo, Japan. He holds a Ph.D. from the University of Western Ontario. His research interests include practical designs of molecular self-assembly systems and combinatorics in general. Being awarded a postdoctoral researcher grant from Academy of Finland, he had served its principal investigator till March 2015, when he assumed his current position in Chofu.

Fig. 1. Single Tile Example

Fig. 2.

(a) An active tile. (b)-(d) The tile modification function acting on a two tile assembly. The initial attachment is shown in (b) and two successive time steps in (c) and (d), with (d) being a completed assembly.

Fig. 3.

a) Input assembly. b) Single step transmission of two activation signals (left to center tile) and a deactivation signal (center to left tile). c) In center tile, deactivation takes place on the right side. In right tile, activation occurs on the right side (top arrow) and another activation signal is received (second arrow from top). Note that this transition from b) to c) yields an unstable assembly map for temperature 1. d) The two stable assemblies corresponding to the maximal stable partition of the assembly map in c). e) The TMF acts separately on the two assemblies in d): the left assembly is unchanged; in the right assembly, a break in the activation chain is detected and the corresponding signal pathway returns into a passive state (initiation signal is removed in second arrow from top).

Fig. 4.

An attachment signal transmission scheme in (i) and a detachment scheme in (ii). The letters a , b , x , y , α , β denote DNA sequences and asterisks indicate the complementary sequences. The system uses branch migration as originally demonstrated by Yurke, et al.¹⁷⁾ and signal transmission as in Padilla, et al.¹⁴⁾

Fig. 5.

Schematic representation of frame tiles. Different glues on the $t_{\pm x}$ sides are indicated positionally. A square inside the tile represents an inactive glue set; each activation and deactivation arrow represents that set of signals in the indicated direction. Active t_y side glue is unique.

Fig. 6.

A schematic representation of the input tiles. Each signal transmission arrow represents the set of activation transmission signals in that direction. There is a unique active glue at the top.

Fig. 7.
A schematic representation of the top computational tiles analogous to the bottom tiles.

Fig. 8.

A schematic representation of the bottom computational tiles analogous to the input tiles with added deactivation sets.

Fig. 9. Attachment and detachment signal transmission scheme between levels on the frame tile. Dashed arrows indicate detachment signals and solid arrows indicate attachment signals.

Fig. 10.

An experimental scheme for attachment and detachment of level tiles. Upon attachment of level 0 tile, the signal is transmitted to detach level 3 tile (not depicted) and to activate a glue for level 1 tile. Attachment of level 1 tile transmits a signal to activate a glue for level 2 tile (not depicted) and a signal that detaches level 0 tile.

Fig. 11. General Initial Frame Sequence

Fig. 12.

Input tiles attach to frame and activate glues for level 1 top computational tiles.

Fig. 13.

Frame with input row signals sent for level 1 top computational tiles. The latter are shown above the $\alpha^{\dot{0}}$ assembly (refer to proof of Proposition 3.1).

Fig. 14.

Frame with level 1 signals sent to deactivate some of the input row glues and activate level 2 glues. Note that the deactivation signals for a given input tile must pass through all three level 1 tiles computed from it before reaching the frame. The detached input row tiles are shown below the α^i assembly.

Fig. 15.

Frame with signals sent from level 2 bottom tiles to deactivate some of the level 1 glues and activate level 3 glues. Note that the three level 2 tiles cannot interact with the adjacent input tiles. A detached level 1 tile is shown above α^2 .

Fig. 16.

Frame with type 3 computed tile (which cannot interact with surrounding level 1 tiles).
Computation cannot proceed without further input.

Fig. 17. Leftward frame extension at the first time step by a signal fired by τ_{x_1} and propagated through τ_{x_0} and input 0 tile.

Table 1

Signal pathway complexity, where the columns Top and Bottom refer to the signal pathway complexity of top computational tiles and bottom computational tiles, respectively. The numbers in parentheses show the numbers for an elementary CA over alphabet $\{0, 1\}$.

	Activation	Deactivation	Act. Trans.	Deact. Trans.
Frame	$4n^3$ (32)	$4n^3 + 1$ (33)	0	0
Input	0	0	$4n^2$ (16)	0
Top	0	0	$4n^2$ (16)	$4n^3 + 4n$ (40)
Bottom	0	0	$4n^2$ (16)	$4n^3$ (32)