Algorithmica (2024) 86:2211–2249
https://doi.org/10.1007/s00453-024-01219-2

The Impacts of Dimensionality, Diffusion, and Directedness on Intrinsic Cross-Model Simulation in Tile-Based Self-Assembly

Daniel Hader[1] · **Matthew J. Patitz**[1]

Received: 9 August 2023 / Accepted: 23 February 2024 / Published online: 3 April 2024

Abstract

Motivated by applications in DNA-nanotechnology, theoretical investigations in algorithmic tile-assembly have blossomed into a mature theory. In addition to computational universality, the abstract Tile Assembly Model (aTAM) was shown to be intrinsically universal (FOCS 2012), a strong notion of completeness where a single tile set is capable of simulating the full dynamics of all systems within the model; however, this construction fundamentally required non-deterministic tile attachments. This was confirmed necessary when it was shown that the class of directed aTAM systems, those where all possible sequences of tile attachments result in the same terminal assembly, is not intrinsically universal (FOCS 2016). Furthermore, it was shown that the non-cooperative aTAM, where tiles only need to match on 1 side to bind rather than 2 or more, is not intrinsically universal (SODA 2014) nor computationally universal (STOC 2017). Building on these results to further investigate the other dynamics, Hader et al. examined several tile-assembly models which varied across (1) the numbers of dimensions used, (2) how tiles diffused through space, and (3) whether each system is directed, and determined which models exhibited intrinsic universality (SODA 2020). In this paper we extend those results to provide direct comparisons of the various models against each other by considering intrinsic simulations between models. Our results show that in some cases, one model is strictly more powerful than another, and in others, pairs of models have mutually exclusive capabilities. This paper is a greatly expanded version of that which appeared in ICALP 2023.

Keywords Tile-assembly · Tiles · ATAM · Intrinsic simulation · Simulation

✉ Daniel Hader
dhader@uark.edu

Matthew J. Patitz
patitz@uark.edu

[1] Department of Computer Science and Computer Engineering, University of Arkansas, Fayetteville, USA

1 Introduction

Self-assembling systems are those in which a disorganized collection of simple components spontaneously combine to form complex, organized structures through random motion and local interactions. From the pristine, periodic arrangements formed by crystallizing atoms to the robust coordination of dividing cells in developing organisms, such systems are the source of much complexity in nature and a topic of critical importance to many fields of research. Among them is the field of DNA nanotechnology, wherein artificial DNA strands are used as structural units that self-assemble according to the dynamics of DNA base pairing, which has seen immense success over the past several decades in harnessing the power of self-assembly to create microscopic structures with incredible precision [1–4] and even perform algorithmic tasks at the nano-scale [5–12]. Because it's difficult and expensive to accurately model the chemistry of DNA, a variety of simplifying models have been proposed to facilitate the design of DNA-based self-assembling systems. Among the more popular and effective ones are tile-assembly (TA) models where components, made of several bound DNA strands exposing small unbound portions with which other components can bind, are abstractly represented as geometric tiles whose labeled sides attach to one another according to predefined affinity rules [13–15]. The advantage of these models lies not only in their success as design tools, but in their similarity to existing models studied heavily in computer science such as Wang tiles and cellular automata. This similarity isn't a coincidence either; the first TA model proposed, the abstract Tile-Assembly Model (aTAM), was designed, at least in part, to show that the dynamics of DNA-based self-assembly are algorithmically universal [15]. Consequently, DNA nanotechnology shares a unique relationship with the theory of computation, with theorists frequently borrowing ideas from complexity, computability, and information theory to study questions regarding, among many other things, what kinds of structures can be self-assembled, the relative difficulty of assembling different shapes, and how variations in a model's dynamics affect its algorithmic power.

This paper is particularly focused on that latter question. As with more conventional models of computation, we generally study such questions by proving whether one model is capable of simulating all systems of another. We have to be careful about our definition of simulation however, as it's generally straightforward to show that many TA models are capable of universal computation. Consequently, most TA models are capable of "simulating" all others in that they can simulate a Turing machine which can in turn simulate the other model. To learn something useful about the relative power of two TA models therefore, we have to consider the geometry of the tile-assembly dynamics. We do this by adapting a tool from the theory of cellular automata, namely *intrinsic simulation*. For a simulation to be *intrinsic*, we require that the simulation is not merely symbolic (i.e. how a Turing machine can simulate an aTAM system by storing an internal representation of the tiles as symbols on its tape), but rather geometric wherein blocks of tiles in the simulating system correspond to individual tiles in the simulated system and the order of tile attachments in these blocks follow those in the simulated system up to a fixed scale factor. In other words, such a simulation would appear identical to the system being simulated if we "zoomed out" sufficiently far (Fig. 1). This approach is not novel to our results, in fact there is already a relatively

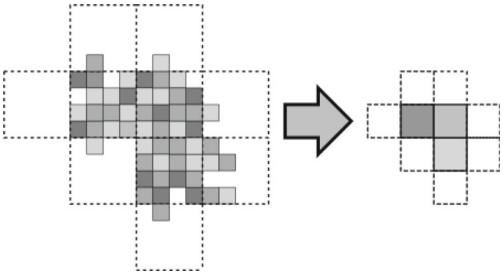

Fig. 1 During an intrinsic simulation, the dynamics of individual tile attachments are simulated so that blocks of tiles in the simulating system "look like" individual tiles at scale

mature theory of intrinsic simulations in tile-assembly which has resulted in a "kind of computational complexity theory for self-assembly" [16]. Such efforts have been instrumental in characterizing the relative power of TA models and has lead to a deeper understanding how different dynamics can be used for the same algorithmic purpose.

1.1 Our Results

In an attempt to extend several previous results regarding intrinsic simulation, here we consider 3 specific variations of the aTAM: *dimensionality*, where both 2D and 3D systems are considered, *diffusion*, where tiles cannot attach in regions which have been surrounded by previously attached tiles, and *directedness*, where tile attachments in a system are required to result in exactly one terminal assembly. It's important to note that these variations aren't arbitrary either. The difference between directed and undirected systems is analogous to the difference between deterministic and probabilistic algorithms and, among other things, plays a role in the study of the complexity of shape assembly [17, 18]. The diffusion restriction on the other hand is often used to make 3D tile-assembly models more realistic by limiting tile attachments to those locations in which a tile could reasonably diffuse (i.e. not in a region completely surrounded by other tiles). There are certainly other variations of the aTAM including those where tiles have negative glues [19, 20], complex geometries [13, 21], and even the ability to propagate signals along their surface [14, 22]; however, we note that these models are generally highly theoretical and have seen little use in the aid of designing physical and practical self-assembling DNA-based tile systems. The 3 variations we have chosen all arise naturally in the design of DNA-based tile systems and are thus well motivated for theoretical comparative study of their relative capabilities.

These variations can be introduced into the aTAM in any combination to yield 8 different models and, considering all ordered pairs of these 8 models gives rise to a table consisting of 64 entries each representing one model's ability or inability to intrinsically simulate the dynamics of another. Generally speaking, results regarding these *cross-model simulations* are complex, involving intricate tile-assembly constructions and counterexamples; consequently, only a handful of these entries have been proved in past literature.

In this paper, we fill a considerable number of missing entries. Table 1 lays out our results along with past results denoted by an asterisk. In it, entries are labeled to indicate whether the model in the row's header can simulate the model in the column's header.

Table 1 Table of our results, outlining whether the row's model can intrinsically simulate the column's model

		aTAM		PaTAM	
		all	dir	all	dir
aTAM	all	Yes* [23]	Yes* [23]	No (Theorem 3, Observation 1)	?
	dir	No (Theorem 1)	No* [24]	No (Theorem 3, Observation 1)	?
PaTAM	all	No (Theorem 2, Observation 1)	No (Theorem 2, Observation 1)	No* [25]	Yes (Theorem 5)
	dir	No (Theorem 2, Observation 1)	No (Theorem 2, Observation 1)	No (Theorem 1)	No* [25]
3DaTAM	all	Yes† (Observation 3)	Yes† (Observation 3)	?	?
	dir	No (Theorem 1)	Yes† (Observation 3)	No (Theorem 1)	?
SaTAM	all	Yes† (Observation 3)	Yes† (Observation 3)	?	?
	dir	No (Theorem 1)	Yes† (Observation 3)	No (Theorem 1)	?

		3DaTAM		SaTAM	
		all	dir	all	dir
aTAM	all	No (Observation 2)	No (Observation 2)	No (Observation 2)	No (Observation 2)
	dir	No (Observation 2)	No (Observation 2)	No (Observation 2)	No (Observation 2)
PaTAM	all	No (Observation 2)	No (Observation 2)	No (Observation 2)	No (Observation 2)
	dir	No (Observation 2)	No (Observation 2)	No (Observation 2)	No (Observation 2)
3DaTAM	all	Yes* [25]	Yes* [25]	No (Theorem 4, Observation 1)	?
	dir	No (Theorem 1)	Yes* [25]	No (Theorem 4, Observation 1)	?
SaTAM	all	Yes† (Observation 4)	Yes† (Observation 4)	Yes* [25]	?
	dir	No (Theorem 1)	Yes† (Observation 4)	No (Theorem 1)	?

PaTAM is the Planar aTAM, 3DaTAM the 3-dimensional aTAM, and SaTAM is the Spatial aTAM (see Sect. 2.2 for full definitions). *All* refers to the set of all systems in a model and *dir* refers to the subset of directed systems. Cells marked with an asterisk (*) are existing results and those marked with a dagger (†) are trivial observations using tile sets from existing results. All other results are novel

There are of course a few entries for which the answer is obvious, which we state as observations with justification rather than full theorems, but many of our results are distinctly non-trivial and some were rather unexpected. For instance, while we initially suspected that the diffusion restricted version of the aTAM (i.e. the Planar aTAM or PaTAM) was, as it's name suggests, a weaker version of the aTAM, we found that both models exhibit dynamics which cannot be simulated by the other. While the table is still missing a few entries, our contributions have brought the number of known entries up to 52 from the 16 which previously existed in published literature (8 of which were technically not explicitly stated, but were trivial observations based on the tile sets and proofs presented in [25]).[1]

A shortened version of this paper was published in [26], and in this version we still include the high-level overview of each result but then also include the full proof details which were omitted in [26]. The rest of our paper is laid out as follows. In Sect. 2, we provide definitions of the various models, concepts, and types of simulation used throughout the paper. In Sect. 3 we present the relatively simple set of results that arise from observations and utilizing prior constructions. In Sect. 4 we prove that there exist aTAM systems that cannot be simulated by any PaTAM systems, and in Sect. 5 we show the reverse, i.e. there exist PaTAM systems that cannot be simulated by any aTAM systems. Thus, those two sections show the mutually exclusive powers of those two models. Section 6 contains our final impossibility result showing that SaTAM systems exist that cannot be simulated by any 3DaTAM systems. In Sect. 7 we present a positive result by construction, showing that there exists a universal PaTAM tile set that can be used to simulate *any* directed PaTAM system. Finally, in Sect. 8 we provide an overview of our work and goals, as well as speculation on possible approaches for solving some of the remaining open problems.

2 Preliminary Definitions

Throughout this paper we will use $\mathbb{Z}$, $\mathbb{Z}^+$, and $\mathbb{N}$ to denote the set of integers, positive integers, and non-negative integers respectively. We will also assume $\mathbb{Z}^d$ has the additional structure of a lattice graph so that each point is a vertex and two points are adjacent (i.e. share an edge) exactly when their Euclidean distance is 1.

2.1 Definition of the Abstract Tile-Assembly Model

In this section, we define the abstract Tile-Assembly Model in 2 and 3 dimensions. We will use the abbreviation *aTAM* to refer to the 2D model and *3DaTAM* for the 3D model. These definitions are borrowed from [25] and we note that [27] is a good introduction to the model for unfamiliar readers.

Fix $d \in \{2, 3\}$ to be the number of dimensions and Σ to be some alphabet with Σ^* its finite strings. A *glue* $g \in \Sigma^* \times \mathbb{N}$ consists of a finite string *label* and non-negative

[1] It should also be noted that most of the remaining unknown entries involve simulating directed, diffusion restricted systems. While we do hope to fill these entries in the future, we suspect that their proofs will be quite complicated since simulating diffusion restricted systems is tricky and counterexamples are often harder to find in directed systems.

integer *strength*. A *tile type* is a tuple $t \in (\Sigma^* \times \mathbb{N})^{2d}$, thought of as a unit square or cube with a glue on each side. A *tile set* is a finite set of tile types. We always assume a finite set of tile types, but allow an infinite number of copies of each tile type to occupy locations in the $\mathbb{Z}^d$ lattice, each called a *tile*.

Given a tile set T, a *configuration* is an arrangement (possibly empty) of tiles in the lattice $\mathbb{Z}^d$, i.e. a partial function $\alpha : \mathbb{Z}^d \dashrightarrow T$. Two adjacent tiles in a configuration *interact*, or are *bound* or *attached*, if the glues on their abutting sides are equal (in both label and strength) and have positive strength. Each configuration α induces a *binding graph* B_α whose vertices are those points occupied by tiles, with an edge of weight s between two vertices if the corresponding tiles interact with strength s. An *assembly* is a configuration whose domain (as a graph) is connected and non-empty. The *shape* $S_\alpha \subseteq \mathbb{Z}^d$ of assembly α is the domain of α. For some $\tau \in \mathbb{Z}^+$, an assembly α is τ*-stable* if every cut of B_α has weight at least τ, i.e. a τ-stable assembly cannot be split into two pieces without separating bound tiles whose shared glues have cumulative strength τ. Given two assemblies α, β, we say α is a *subassembly* of β (denoted $\alpha \sqsubseteq \beta$) if $S_\alpha \subseteq S_\beta$ and for all $p \in S_\alpha$, $\alpha(p) = \beta(p)$.

A *tile-assembly system* (TAS) is a triple $\mathcal{T} = (T, \sigma, \tau)$, where T is a tile set, σ is a finite τ-stable assembly called the *seed assembly*, and $\tau \in \mathbb{Z}^+$ is called the *binding threshold*. Given a TAS $\mathcal{T} = (T, \sigma, \tau)$ and two τ-stable assemblies α and β, we say that α $\mathcal{T}$*-produces* β *in one step* (written $\alpha \to_1^{\mathcal{T}} \beta$) if $\alpha \sqsubseteq \beta$ and $\left|S_\beta \backslash S_\alpha\right| = 1$. That is, $\alpha \to_1^{\mathcal{T}} \beta$ if β differs from α by the addition of a single tile. The $\mathcal{T}$*-frontier* is the set $\partial^{\mathcal{T}} \alpha = \bigcup_{\alpha \to_1^{\mathcal{T}} \beta} S_\beta \backslash S_\alpha$ of locations in which a tile could τ-stably attach to α.

We use $\mathcal{A}^T$ to denote the set of all assemblies of tiles in tile set T. Given a TAS $\mathcal{T} = (T, \sigma, \tau)$, a sequence of $k \in \mathbb{Z}^+ \cup \{\infty\}$ assemblies $\alpha_0, \alpha_1, \ldots$ over $\mathcal{A}^T$ is called a $\mathcal{T}$*-assembly sequence* if, for all $1 \leq i < k$, $\alpha_{i-1} \to_1^{\mathcal{T}} \alpha_i$. The *result* of an assembly sequence is the unique limiting assembly of the sequence. For finite assembly sequences, this is the final assembly; whereas for infinite assembly sequences, this is the assembly consisting of all tiles from any assembly in the sequence. We say that α $\mathcal{T}$*-produces* β (denoted $\alpha \to^{\mathcal{T}} \beta$) if there is a $\mathcal{T}$-assembly sequence starting with α whose result is β. We say α is $\mathcal{T}$*-producible* if $\sigma \to^{\mathcal{T}} \alpha$ and write $\mathcal{A}[\mathcal{T}]$ to denote the set of $\mathcal{T}$-producible assemblies. We say α is $\mathcal{T}$*-terminal* if α is τ-stable and there exists no assembly which is $\mathcal{T}$-producible from α. We denote the set of $\mathcal{T}$-producible and $\mathcal{T}$-terminal assemblies by $\mathcal{A}_\square[\mathcal{T}]$.

When $\mathcal{T}$ is clear from context, we may omit $\mathcal{T}$ from the notation above.

Cooperative Attachment

Given a TAS $\mathcal{T} = (T, \sigma, \tau)$, for a tile to attach to an assembly it must match glues whose cumulative strength is at least τ in order to result in a τ-stable assembly. This can happen if, for instance, one of the matched glues has strength at least τ, in which case any other matching glues are superfluous. Alternatively, a tile may still attach without any τ-strength glues though this requires multiple glues to match whose strengths sum to at least τ. We refer to such attachments as *cooperative*.

2.2 Model Variations

In this paper we consider 3 variations of the aTAM. Other than the 3D aTAM, these include *directed* and *diffusion restricted* versions of the models. We say that a TAS $\mathcal{T}$ is *directed* if $|\mathcal{A}_{\square}[\mathcal{T}]| = 1$, i.e. $\mathcal{T}$ admits only a single producible terminal assembly. When we refer to a *directed model* we simply mean the set of all directed systems in a model. Directed systems are desirable for self-assembly since we often want our tiles to grow into a single target shape.

For diffusion restricted models, we note that in the aTAM it's possible for tiles to attach within a region of space which has been completely surrounded by other tiles. In 2D, we can imagine that the tiles are able to navigate around the assembly through the 3rd dimension, but in 3D such attachments are difficult to justify. Consequently, we also consider models where such attachments are forbidden. In 2D, this restriction could model a self-assembly process on the surface of a droplet of water where surface tension prevents the components from taking advantage of the 3rd dimension. We call the 2D diffusion restricted aTAM the *Planar aTAM* or PaTAM, and we call the 3D diffusion restricted aTAM the *Spatial aTAM* or SaTAM. In these models, and their directed subsets, we refer to regions which have been completely surrounded (in which no tile attachments are allowed to occur) *constrained*. To formally model this restriction, we first note that given a finite d-dimensional assembly α, the graph $\mathbb{Z}^d \setminus S_\alpha$ consists of a finite number of connected components, exactly one of which will be infinite in size. We say that this component graph is the *outside* of α while the finite-sized components are *constrained*. In a diffusion restricted system we only allow tile attachments on the outside of an assembly.

2.3 Intrinsic Simulation

First we provide a high-level definition of the notion of *intrinsic simulation* which should be sufficient for understanding our results. A full technical definition follows afterward. For brevity, in this paper, unless explicitly stated, "simulation" will refer to intrinsic simulation.

High-Level Description of Simulation

Simulation of system $\mathcal{T}$ by system $\mathcal{S}$ occurs at a scale factor m, so that $m \times m$ (or $m \times m \times m$ in 3D) blocks of tiles from $\mathcal{S}$, which we refer to as *macrotiles*, correspond to individual tiles in $\mathcal{T}$. For a given simulation, we define a *macrotile representation function* R which describes this mapping of macrotiles to tiles. Additionally for convenience, using R we define an *assembly representation function* R^* which maps entire assemblies from $\mathcal{S}$ to assemblies in $\mathcal{T}$, essentially evaluating R on each macrotile location for a given assembly in $\mathcal{S}$. Note that we don't require all locations within a macrotile to contain a tile and macrotile blocks containing tiles can still be mapped to empty space under R. When a tile attachment causes the representation of a macrotile location to map to a tile for the first time, we say that the attachment has caused the macrotile to *resolve* and once a macrotile has resolved, any additional tile attachments within the macrotile cannot change its representation under R. While we do allow macrotile locations to map to empty space, for a simulation to be valid there must be

restrictions on where tiles are allowed to attach in $\mathcal{S}$. For our notion of simulation to be useful as a metric of comparing the relative capabilities of models, we require that $\mathcal{S}$ only place tiles within the macrotile regions immediately adjacent (not diagonally) to those which have already resolved, and we call such locations *fuzz*. This allows tiles in $\mathcal{S}$ to attach only in macrotiles which could potentially resolve during a valid simulation, since only the locations in $\mathcal{T}$ mapped to by the fuzz locations could possibly receive tiles in $\mathcal{T}$. If a class of systems C can all be simulated by another class of systems C' sharing a single tile set (though each may have a different seed assembly), we say that class C' *intrinsically simulates* C with a *universal tile set*. We can also say that C' is *intrinsically universal* (IU) for C.

Formal Definition of Simulation

Now we provide formal definitions for *intrinsic simulation*. The definitions here are taken from [25] and specifically refer to 3D systems. Similar definitions for 2D intrinsic simulation are given in [24]. For simulation of a 2D system by a 3D system, we use the 3D definitions and assume that all systems in the 2D system are defined in 3D so that assemblies occupy only the $z = 0$ plane.

From this point on, let T be a tile set and let the scale factor be $m \in \mathbb{Z}^+$. An *m-block macrotile* over T is a partial function $\alpha : \mathbb{Z}_m^3 \dashrightarrow T$, where $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$. Let B_m^T be the set of all m-block macrotiles over T. The m-block with no domain is said to be *empty*. For a general assembly $\alpha : \mathbb{Z}^3 \dashrightarrow T$ and $(x', y', z') \in \mathbb{Z}^3$, define $\alpha^m_{(x',y',z')}$ to be the m-block macrotile defined by $\alpha^m_{(x',y',z')}(i_x, i_y, i_z) = \alpha(mx' + i_x, my' + i_y, mz' + i_z)$ for $0 \leq i_x, i_y, i_z < m$. For some tile set S, a partial function $R : B_m^S \dashrightarrow T$ is said to be a *valid m-block macrotile representation* from S to T if for any $\alpha, \beta \in B_m^S$ such that $\alpha \sqsubseteq \beta$ and $\alpha \in \text{dom } R$, then $R(\alpha) = R(\beta)$.

For a given valid m-block macrotile representation function R from tile set S to tile set T, define the *assembly representation function*[2] $R^* : \mathcal{A}^S \to \mathcal{A}^T$ such that $R^*(\alpha') = \alpha$ if and only if $\alpha(x, y, z) = R\left(\alpha'^m_{(x,y,z)}\right)$ for all $(x, y, z) \in \mathbb{Z}^3$. For an assembly $\alpha' \in \mathcal{A}^S$ such that $R^*(\alpha') = \alpha$, α' is said to map *cleanly* to $\alpha \in \mathcal{A}^T$ under R^* if for all non empty blocks $\alpha'^m_{(x,y,z)}$, $(x, y, z) + (u_x, u_y, u_z) \in \text{dom}\,(\alpha)$ for some $(u_x, u_y, u_z) \in U_3$ such that $u_x^2 + u_y^2 + u_z^2 \leq 1$, or if α' has at most one non-empty m-block $\alpha^m_{0,0}$. In other words, α' may have tiles on macrotile blocks representing empty space in α, but only if that position is adjacent to a tile in α. We call such growth "around the edges" of α' *fuzz* and thus restrict it to be adjacent to only valid macrotiles, but not diagonally adjacent (i.e. we do not permit *diagonal fuzz*).

In the following definitions, let $\mathcal{T} = (T, \sigma_T, \tau_T)$ be a TAS, let $\mathcal{S} = (S, \sigma_S, \tau_S)$ be a TAS, and let R be an m-block representation function $R : B_m^S \to T$.

Definition 1 We say that $\mathcal{S}$ and $\mathcal{T}$ have *equivalent productions* (under R), and we write $\mathcal{S} \Leftrightarrow \mathcal{T}$ if the following conditions hold:

1. $\left\{R^*(\alpha') | \alpha' \in \mathcal{A}[\mathcal{S}]\right\} = \mathcal{A}[\mathcal{T}]$.
2. $\left\{R^*(\alpha') | \alpha' \in \mathcal{A}_\square[\mathcal{S}]\right\} = \mathcal{A}_\square[\mathcal{T}]$.
3. For all $\alpha' \in \mathcal{A}[\mathcal{S}]$, α' maps cleanly to $R^*(\alpha')$.

[2] Note that R^* is a total function since every assembly of S represents *some* assembly of T; the functions R and α are partial to allow undefined points to represent empty space.

Definition 2 We say that $\mathcal{T}$ *follows* $\mathcal{S}$ (under R), and we write $\mathcal{T} \dashv_R \mathcal{S}$ if $\alpha' \to^{\mathcal{S}} \beta'$, for some $\alpha', \beta' \in \mathcal{A}[\mathcal{S}]$, implies that $R^*(\alpha') \to^{\mathcal{T}} R^*(\beta')$.

The next definition essentially specifies that every time $\mathcal{S}$ simulates an assembly $\alpha \in \mathcal{A}[\mathcal{T}]$, there must be at least one valid growth path in $\mathcal{S}$ for each of the possible next steps that $\mathcal{T}$ could make from α which results in an assembly in $\mathcal{S}$ that maps to that next step. While this definition is unfortunately dense, it accommodates subtle situations such as where $\mathcal{S}$ must "commit to" a subset of possible representations in $\mathcal{T}$ before being explicitly mapped, under R, to any one in particular.

Definition 3 We say that $\mathcal{S}$ *models* $\mathcal{T}$ (under R), and we write $\mathcal{S} \models_R \mathcal{T}$, if for every $\alpha \in \mathcal{A}[\mathcal{T}]$, there exists $\Pi \subset \mathcal{A}[\mathcal{S}]$ where $\Pi \neq \emptyset$ and $R^*(\alpha') = \alpha$ for all $\alpha' \in \Pi$, such that, for every $\beta \in \mathcal{A}[\mathcal{T}]$ where $\alpha \to^{\mathcal{T}} \beta$, (1) for every $\alpha' \in \Pi$ there exists $\beta' \in \mathcal{A}[\mathcal{S}]$ where $R^*(\beta') = \beta$ and $\alpha' \to^{\mathcal{S}} \beta'$, and (2) for every $\alpha'' \in \mathcal{A}[\mathcal{S}]$ where $\alpha'' \to^{\mathcal{S}} \beta'$, $\beta' \in \mathcal{A}[\mathcal{S}]$, $R^*(\alpha'') = \alpha$, and $R^*(\beta') = \beta$, there exists $\alpha' \in \Pi$ such that $\alpha' \to^{\mathcal{S}} \alpha''$.

In this definition, Π is a set of assemblies in $\mathcal{S}$ which map to a given assembly α in $\mathcal{T}$ under the representation function R^*. Specifically, this set Π represents the assemblies in $\mathcal{S}$ which are still capable of resolving into any assembly producible from α in $\mathcal{T}$ (this is condition 1 in the definition). Furthermore, condition 2 stipulates that while any assembly in $\mathcal{S}$ representing α may have already grown to the point where it is no longer possible to resolve into all assemblies in $\mathcal{T}$ producible from α, it must have been possible at some previous time during the assembly sequence for all assemblies producible from α to be represented.

Definition 4 We say that $\mathcal{S}$ *intrinsically simulates* $\mathcal{T}$ (under R) if $\mathcal{S} \Leftrightarrow_R \mathcal{T}$ (equivalent productions), $\mathcal{T} \dashv_R \mathcal{S}$ and $\mathcal{S} \models_R \mathcal{T}$ (equivalent dynamics).

2.4 Window Movie Lemma

In [28], the authors proved the Window Movie Lemma, a pumping lemma of sorts for the aTAM (and its variants) which has since seen much use as a powerful tool for proving that certain tile-assembly simulations are impossible. Since it appears in several of our proofs, we first informally describe the lemma, then explicitly state it. A *window* is an edge cut which partitions the lattice graph ($\mathbb{Z}^2$ in 2D or $\mathbb{Z}^3$ in 3D) into two regions. Given some window w and some assembly sequence $\vec{\alpha}$ in a TAS $\mathcal{T}$, a *window movie* M is defined to be the ordered sequence of glues presented along w by tiles in $\mathcal{T}$ during the assembly sequence $\vec{\alpha}$. Informally, if we think of the window w as a thin pane dividing two regions of tile locations and imagine stepping through the assembly sequence $\vec{\alpha}$ one tile attachment at a time, M is constructed by recording the glues which appear on the surface of the pane and their relative order. More formally, a *window movie* is the sequence $M_w^{\vec{\alpha}} = \{(v_i, g_i)\}$ of pairs of grid graph vertices v_i and glues g_i, given by order of appearance of the glues along window w during $\vec{\alpha}$. Furthermore, if k glues appear along w during the same assembly step in $\vec{\alpha}$, then these glues appear contiguously and are listed in lexicographical order of the unit vectors describing their orientation in $M_w^{\vec{\alpha}}$.

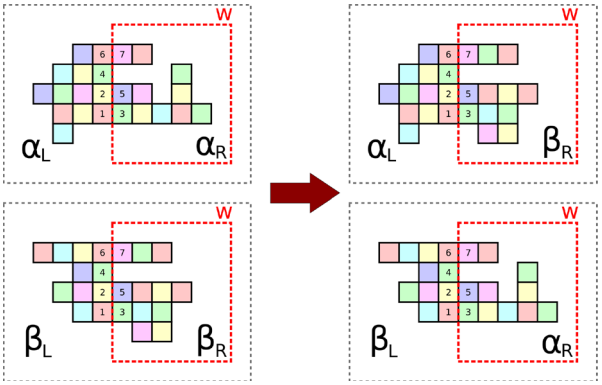

Fig. 2 An illustration of the window movie lemma. On the left are two producible assemblies $\alpha = \alpha_L \cup \alpha_R$ and $\beta = \beta_L \cup \beta_R$ made from the same tile set, which are each divided into two subassemblies by the window w. For both assemblies, the window w has the same window movie, i.e. the order in which tiles present glues along the window, depicted by numbers on the tiles describing the relative order in which they attached. Since all growth within the windowed regions depends only on the glues presented along the window, we can splice these assemblies to get $\alpha_L \cup \beta_R$ or $\beta_L \cup \alpha_R$ (illustrated on the right). The window movie lemma then guarantees that both of these assemblies are producible

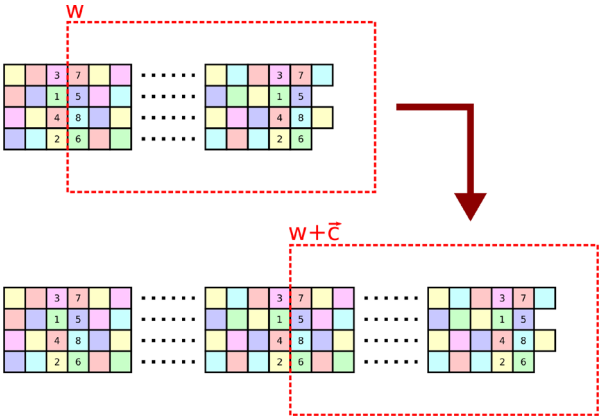

Fig. 3 Using the Window Movie Lemma to "pump" assembly sequences. The top assembly depicts a ribbon of tiles growing horizontally to the right and numbers on tiles describe a relative order of attachment. If such a ribbon of tiles grows long enough, then by pigeonhole principle, eventually there must exist two identical vertical slices along its length. Because every tile attachment inside a window w depends only on the tiles and their relative order of attachment along the window, we can thus find an assembly sequence where growth repeats after the second identical vertical slice. This can be performed indefinitely to "pump" the ribbon

Informally, the Window Movie Lemma states that any tile attachments that occur within the region bounded by a window are possible in a region bounded by the same window (up to translation) with an identical window movie (Fig. 2). This allows us to splice assembly sequences together and, consequently, pump a sequence of tile attachments so long as we can ensure the existence of identical window movies. Figure 3 illustrates how the Window Movie Lemma can be used to pump growth.

Window Movie Lemma
Let $\vec{\alpha} = \{\alpha_i\}$ and $\vec{\beta} = \{\beta_i\}$ be assembly sequences in TAS $\mathcal{T}$ and let α, β be the result assemblies of each respectively. Let w be a window that partitions α into two configurations α_L and α_R and let $w' = w + \vec{c}$ be a translation of w that partitions β into two configurations β_L and β_R (with α_L and β_L being the configurations containing their respective seed tiles). Furthermore define $M_w^{\vec{\alpha}}$ and $M_w^{\vec{\beta}}$ to be the window movies for $\vec{\alpha}, w$ and $\vec{\beta}, w'$ respectively. Then if $M_w^{\vec{\alpha}} = M_w^{\vec{\beta}}$, the assemblies $\alpha_L \cup \beta'_R$ and $\beta'_L \cup \alpha_R$ (where $\beta'_L = \beta_L - \vec{c}$ and $\beta'_R = \beta_R - \vec{c}$) are also producible.

3 Observations and Simpler Results

In this section we present some relatively trivial observations that allow us to fill in several boxes from Table 1.

Observation 1 *If there exists a system $\mathcal{T}$ in the directed subset of systems in tile-assembly model M which cannot be simulated by any system in tile-assembly model M', then (1) there exists a system in M which cannot be simulated by any system in M', (2) there exists a system in M which cannot be simulated by any directed system in M', and (3) there exists a directed system in M which cannot be simulated by any directed system in M'.*

Observation 2 *There exists systems, both directed and undirected, in the 3D models (3DaTAM and SaTAM) which cannot be simulated by any systems in any of the 2D models (aTAM and PaTAM, both directed and undirected).*

Observation 1 holds because the set of directed systems in a model is a subset of all systems in that model. Consequently, $\mathcal{T}$ is a system in both M and in the directed subset of M. By assumption, $\mathcal{T}$ cannot be simulated by any system in M' and therefore cannot be simulated by any subset of systems of M', particularly the subset of directed systems. Regarding Observation 2, while we restrict the notion of simulation to use square macrotiles, simulations of systems on triangular lattices have been implemented using roughly hexagonal macrotiles made from square tiles [29], so one might imagine the possibility that by loosening our definition of simulation to use more interesting macrotiles, it could be possible to capture the geometry of 3D square tiles using 2D tiles. In our case however, we note that there can exist no planar embedding of the lattice graph of $\mathbb{Z}^3$ as a consequence of Kuratowski's theorem. Consequently, there can be no way to divide $\mathbb{Z}^2$ into connected regions of macrotile locations which preserves the adjacency of points in $\mathbb{Z}^3$ and therefore simulation could not be possible even if we generalized our notion of macrotiles. This is true for any 3D systems which have producible assemblies whose domains, as graphs, are non-planar as is trivially possible in all 3D models considered.

3.1 Simulations Using Existing Tile Sets

In [25], it was shown that there exists IU tile sets for the 3DaTAM, SaTAM, and both models' subsets of directed systems. While the main focus of that result was intrinsic

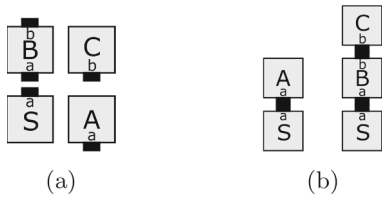

Fig. 4 **a** Tile set of an undirected system for the proof of Theorem 1 and **b** its two terminal assemblies

simulation within a model, those IU tile sets can be used to trivially fill in a few boxes of Table 1. First we note that any aTAM system can also be thought of as a 3DaTAM system (or even SaTAM system since tiles occupying only a single plane of 3D space can't constrain a 3D region) with glues only appearing on 4 of the 6 faces of any tile. Second, we note that the IU tile sets for the 3DaTAM and SaTAM differed only by the addition of a few tile types responsible for growing a wall around each face of a macrotile before resolving. This was necessary for intrinsic universality in the SaTAM since without them, the tiles making up a macrotile were sparse enough to necessarily allow a diffusion path for tiles to pass through a resolved macrotile. Consequently, if we don't include those tile types, then the IU tile set can simulate 3DaTAM systems even in the SaTAM since without walls surrounding each macrotile, the diffusion restriction does not interfere with the attachment of any tiles. Finally, by design, this tile set preserves directedness when simulating a directed system. Therefore, using the IU tile set and proofs from [25], the following observations hold.

Observation 3 *There exists a universal tile set in both the 3DaTAM and SaTAM which intrinsically simulates all systems in the aTAM, preserving directedness.*

Observation 4 *There exists a universal tile set in the SaTAM which intrinsically simulates all systems in the 3DaTAM, preserving directedness.*

3.2 Directed Systems Cannot Simulate Undirected Systems

Theorem 1 *There exist systems in the aTAM, 3DaTAM, PaTAM, and SaTAM, which cannot be simulated by any directed system in any of these models.*

High-level overview.
Whereas directed systems only have one terminal assembly, undirected systems can have several. Figure 4 illustrates the tile set and terminal assemblies of a simple undirected system $\mathcal{T}$ which can be a system in the aTAM, 3DaTAM, PaTAM, or SaTAM without modification as it does not use any dynamics unique to any of those models. Because directed systems can only have a single terminal assembly, any directed system attempting to simulate $\mathcal{T}$ would necessarily fail since any assembly representation function R^* could not map one terminal assembly to both terminal assemblies of $\mathcal{T}$.

Proof Let T be the tile set shown in Fig. 4a. Define a tile assembly system $\mathcal{T} = (T, \sigma, 1)$ where σ consists of a single tile: a copy of the tile labeled S located at the origin. Note that if $\mathcal{T}$ is a system in the aTAM, 3DaTAM, PaTAM, or SaTAM, it

behaves identically: it is an undirected system with exactly two terminal assemblies, which are shown in Fig. 4b.

We prove Theorem 1 by contradiction. Therefore, assume that $\mathcal{S} = (S, \sigma', \tau)$ is a directed system which simulates $\mathcal{T}$ (under representation function R and at scale factor c). By the definition of simulation, $R(\sigma') = \sigma$ (i.e. the seed σ' represents σ), and there exists at least one assembly sequence in $\mathcal{S}$ such that in the resulting terminal assembly the $c \times c$ macrotile region north of σ' represents, under R, the tile of T labeled A, and there also exists at least one assembly sequence in $\mathcal{S}$ such that in the resulting terminal assembly the $c \times c$ macrotile region north of σ' represents, under R, the tile of T labeled B. However, since $\mathcal{S}$ is directed, there can only be one terminal assembly, and therefore both assembly sequences result in the exact same tiles being placed in that $c \times c$ macrotile region. But, since R is a function, it cannot map the same input macrotile region to two different tiles. This is a contradiction, and therefore $\mathcal{S}$ does not simulate $\mathcal{T}$, and since the only assumption made about $\mathcal{S}$ was that it was directed, no directed system can simulate $\mathcal{T}$ (whether it is in the aTAM, 3DaTAM, PaTAM, or SaTAM) and Theorem 1 is proven. □

4 The PaTAM Cannot Simulate the aTAM

Here we show that there are aTAM systems which cannot be correctly simulated by any PaTAM systems. To show this, we take advantage of the fact that aTAM systems are capable of growth inside of constrained regions while PaTAM systems are not. Specifically, we show that the PaTAM can't simulate the directed aTAM and, by Observation 1, note that this also implies that the PaTAM can't simulate the aTAM.

Theorem 2 *There exists a system $\mathcal{T}$ which is a directed aTAM system, and therefore also an aTAM system, which cannot be simulated by any PaTAM system.*

High-level Overview

Figure 5 is a schematic diagram of the terminal assembly of $\mathcal{T}$, a directed aTAM system which we claim is impossible to simulate in the PaTAM. Note that it is not sufficient to simply chose $\mathcal{T}$ to be a system where a tile attaches within a single potentially constrained region. This is because the definition of intrinsic simulation allows for macrotiles to resolve even when they aren't completely filled with tiles. Consequently, while macrotiles may map to tiles constraining a region, the tiles making up the macrotiles may not constrain a region. Our construction is designed to

ensure that at some point, any supposed simulating system must constrain a region before the tiles inside are able to attach. In our directed aTAM system $\mathcal{T}$, this is done by first initiating the growth of a *planter*, a gadget that counts up in binary as it grows eastward, initiating the growth of increasingly tall arms at defined intervals. These arms are essentially binary counter gadgets which each grow upward to a distance, encoded in the glues of the tiles provided by the planter, and initiate the growth of thin arms when they finish. The thin arms are just a single tile wide and begin by growing a fixed distance to the west before growing south to crash into the planter below. By this process, each arm initiated by the planter constrains increasingly large

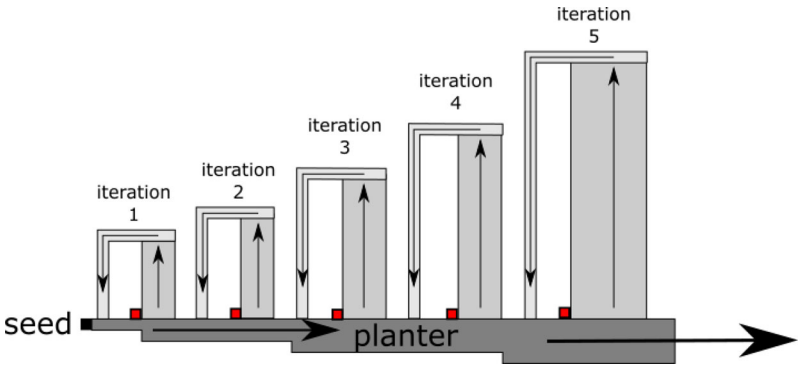

Fig. 5 The aTAM system $\mathcal{T}$ of the proof of Theorem 2. An infinite planter grows to the east from the seed and initiates upward growth of an infinite series of counters, each taller than the last, which initiate single-tile-wide paths that grow to the left then crash downward into the planter. To the left of each counter, at its base, it is possible for a red tile to attach

regions of space which each contain a single location between the planter and arms, in which a single tile can cooperatively attach (denoted by the red squares in Fig. 5). Each of the tiles making up the southward growing portion of the thin arms are of the same tile type, each with identical glues on their north and south faces. While it is possible for different macrotiles to map to the same tile in $\mathcal{T}$, there are only so many combinations of tiles that make up a macrotile. Consequently, regardless of scale factor, if we look far enough down the planter, there will be an arm which grows tall enough that the simulating set must repeat a macrotile representation in two places along the same thin arm. We can then use the Window Movie Lemma to show that this arm "pumps" in our supposed simulating system, before crashing into the planter. It is therefore impossible for any simulating PaTAM system to prevent a region from becoming constrained before the macrotile inside is able to resolve, yielding terminal assemblies which aren't correctly mapped to a terminal assembly in $\mathcal{T}$.

Proof Let $\mathcal{T}$ be the system which is schematically depicted in Fig. 5. $\mathcal{T}$ has a seed consisting of a single tile placed at $(0, 0)$ and has a binding threshold of 2 (i.e. $\tau = 2$). From the seed tile, $\mathcal{T}$ grows a "planter", which is simply a modified log-width binary counter which counts from 8 to ∞. It is modified so that as it counts each number $8 \leq n < \infty$, the bits of that n are rotated upward so that they can initiate a counter which grows upward n rows. Additionally, there are 8 extra spaces between the bits of each counter. The planter grows infinitely to the right, independently of each upward growing counter which it initiates. Each upward growing counter, seeded with the bits of a unique value of n, grows upward n rows. At that point, it grows a single additional row across the top and then a single-tile-wide arm 4 tiles to the left, which then allows a tile to attach to its south which has τ-strength glues allowing copies of itself to attach to its north and south. This results in a single-tile-wide column composed of copies of that tile which grows downward until it eventually "crashes" into the planter (i.e. a tile of the column is placed adjacent to the planter so that no additional tiles can be placed). Note that the growth of the planter is designed so that the location of the planter into which a downward growing arm crashes must be completed before the

counter which eventually initiates the growth of that arm can begin. Finally, at any time after the growth of the first row of the upward growing counter a red tile can bind to the leftmost tile of that row.

By careful design of the modules of $\mathcal{T}$ (which are standard modules in many aTAM constructions, see [24, 30], e.g.), it is clear that $\mathcal{T}$ is not only a valid aTAM system, but it is also directed. We prove Theorem 2 by contradiction, so therefore assume that $\mathcal{P}$ is a PaTAM system with tileset P which simulates $\mathcal{T}$. Let c be the scale factor by which $\mathcal{P}$ simulates $\mathcal{T}$, and let R be the representation function.

For each $8 \leq n < \infty$, we use the term "nth *iteration*" to refer to the growth of the upward counter to n, the arm which grows to the left then downward, and the red tile associated with that n. Notice that the downward growing arms become arbitrarily tall, but remain a constant width. By the definition of simulation and inspection of $\mathcal{T}$, we see that number of tiles spanning any row of an arm cannot be more than $3c$, which is the width of 3 macrotiles. This number includes the macrotile representing the tile of the arm, plus one macrotile of fuzz on each side. Anything outside that width would violate the constraints on fuzz in the simulation and make the simulation invalid. Therefore, let $p = (3c)!(|P|+1)^{3c}$ and note that this is an upper bound on the number of orders in which tiles from $|P|$ (including the lack of a tile) can be placed in a row of $3c$ tile locations.

Now, consider some assembly sequence $\vec{\alpha}$ which grows the assembly up to the $2p$th iteration where the counter grows to a height of $2p$, but where the $2p$th red tile has not yet attached. Since by assumption $\mathcal{P}$ simulates $\mathcal{T}$, there must be a corresponding assembly sequence $\vec{\alpha}'$ in $\mathcal{T}$. Notice that by our definition of p and the size of the iteration, tiles must attach in the same way and order on the top rows of at least two distinct macrotiles (and surrounding fuzz) corresponding to the downward growing arm. We can then define two windows w_1 and w_2, both as the boundary of a $3c \times p+1$ rectangle with the tops centered along the top rows of these macrotiles and note that during $\vec{\alpha}'$ both windows will have the same window movies. Consequently we can construct a new assembly sequence $\vec{\beta}'$ in $\mathcal{T}$, similar to $\vec{\alpha}'$ except that tiles attach identically in the regions enclosed by both windows. These tile attachments can then be repeated until blocked by some tile in the planter macrotiles or surrounding fuzz. At this point, since $\mathcal{P}$ is a Planar aTAM system, it is impossible for tiles to attach in the macrotile region representing the red tile for that iteration. Regardless of how tiles attach after this, the macrotile corresponding to the red tile will never be able to resolve and thus the simulation will be incorrect. Since, $\mathcal{P}$ fails to simulate $\mathcal{T}$, and since we made no assumptions about $\mathcal{P}$ other than the fact that it is a PaTAM system which simulates $\mathcal{T}$, no such simulator can exist. □

5 The aTAM Cannot Simulate the PaTAM

Given that the PaTAM is just the aTAM with an added restriction on tile attachment, it's not terribly surprising that the PaTAM can't simulate the full dynamics of the aTAM; however, less obvious is the fact that the planarity restriction *also* gives the PaTAM some capabilities not possible in the aTAM, namely the ability to constrain a

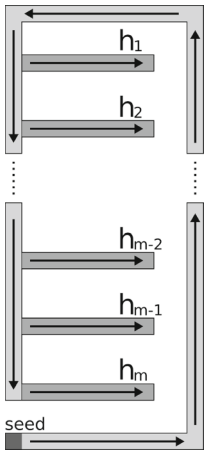

Fig. 6 A schematic of the PaTAM system $\mathcal{P}$ for the proof of Theorem 3. Tiles grow in a rectangular shape, periodically spawning arms which can crash into the walls and constrain a region. It is undirected and its size depends non-deterministically on the number of tiles that attach between each corner

region and stop growth within. We utilize this ability in our proof of Theorem 3. Also, by Observation 1, this also holds for the directed aTAM.

Theorem 3 *There exists a PaTAM system $\mathcal{P}$ which cannot be simulated by any aTAM system.*

High-level Overview

As with the proof for Theorem 2, in the definition of intrinsic simulation, we consider all possible representation functions and scale factors to prove impossibility. Figure 6 is a schematic diagram of the PaTAM system $\mathcal{P}$ which we show is impossible to correctly simulate in the aTAM. Growth of $\mathcal{P}$ begins with tiles attaching in a row growing east. The length of this row is non-deterministic as at any point along the row, it's possible for a corner tile to attach, initiating growth to the north. Consequently, $\mathcal{P}$ is an undirected system so any potential simulating system must be able to simulate all possible assemblies of $\mathcal{P}$. Similarly, northward and eastward growing rows of tiles attach with some length depending on how many tiles attached before each corner. Finally, a column of tiles begins growing south and, as it does, initiates the growth of several arms eastward, each spaced 4 tiles apart. Both the southward growing column of tiles and the arms continue growth until they are constrained or crash into another part of the assembly. To show that $\mathcal{P}$ cannot be simulated in the aTAM, we assume the existence of a simulating aTAM system $\mathcal{T}$ and prove that it must admit some assembly sequences which don't correspond to those in $\mathcal{P}$. To do this, we consider an assembly sequence in $\mathcal{P}$ where the rectangle of tiles grows to a size, based on the scale factor of the simulation, so that a sufficiently large number of sufficiently long arms are spawned by the south growing column of tiles. We also choose an assembly sequence where the south growing column will eventually collide with the seed tile, constraining the region containing the arms. Because we've chosen the assembly to be sufficiently large, each arm is capable of being "pumped" as per the window movie lemma. We then grow the bottom arm until just after it has collided with the east wall and note that, while $\mathcal{T}$ is an aTAM system and can still grow tiles inside of the

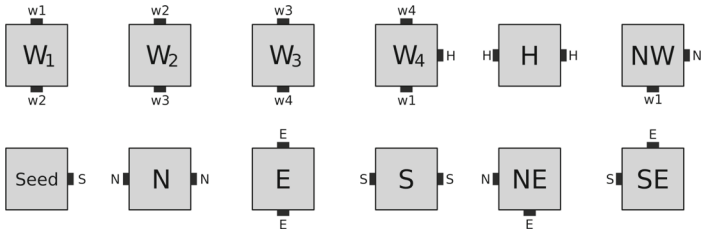

Fig. 7 Tileset for a PaTAM system which cannot be correctly simulated by any aTAM system

constrained region, tiles on the inside and outside will no longer be able to affect each other's growth. There are a few cases to be considered, depending on whether or not the representation function has resolved the last tile of the bottom arm, but essentially we then show that we can continue the growth of the west wall until its macrotiles have resolved to tiles in $\mathcal{P}$ that constrain the rectangle's interior. By a counting argument and our choice of the number of arms, we can then show that one of the other arms must be able to continue growth within the constrained region, and that the assembly sequence in $\mathcal{T}$ maps to one invalid in $\mathcal{P}$.

Proof To prove Theorem 3, let $\mathcal{P}$ be the PaTAM system whose tile types are shown in Fig. 7 and whose growth is illustrated in Fig. 8. This system starts with a single seed tile from which a rectangular frame grows. The south, east, and north walls of this frame are each made of several copies of a single tile type unique to that side, while the west wall is made of 4 distinct tile types which grow in a periodic sequence. Because the frame's first 3 sides each consist of copies of a single tile type, the length of the each side is non-deterministic and depends on how many copies of each type happen attach before a corner tile. Additionally, there is no corner tile which attaches after the west wall tiles so it will grow indefinitely or until it collides with another tile. As the west wall grows, each 4th tile initiates the eastward growth of an arm which grows by the attachment of identical tiles indefinitely or until the region is constrained or a collision occurs. All glues in this system are strength-1 and the binding threshold is likewise 1.

We show that no system in the aTAM is capable of simulating all assembly sequences of $\mathcal{P}$ by contradiction. Therefore, assume that $\mathcal{T} = (T, \sigma, \tau)$ is an aTAM system which simulates $\mathcal{P}$, let c be the scale factor, and let R be the representation function of the simulation. Now, let $p = (3c)! \cdot (2(g+1))^{3c}$. This is an upper bound on the number of orders in which tiles attachments from T (including the lack of a tile) be placed in along the boundary between 2 rows of $3c$ tile locations. Consequently, p bounds the number of possible 1D slices of scale-c macrotiles, including two adjacent fuzz macrotile regions, accounting for the relative order in which the tiles attach. This can be thought of as a pumping length of sorts since, if a line of at least p identical tiles is growing in $\mathcal{P}$, sufficiently far enough away from other tiles, then it must be the case that, between at least two of the corresponding macrotiles in $\mathcal{T}$, an identical sequence of tile attachments occurred. When this happens, it's then possible to consider an assembly sequence wherein any tile attachments beyond the second of these identical macrotile boundaries mimics the attachments beyond the first. This implies

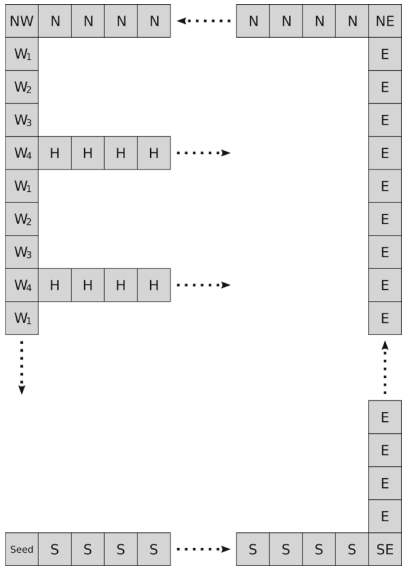

Fig. 8 This PaTAM system cannot be correctly simulated by any aTAM system

the existence of assembly sequences in $\mathcal{T}$ with periodic growth which can continue indefinitely or until blocked by other parts of the assembly. □

Consider now an assembly α_0 in $\mathcal{P}$ wherein the backward C shaped frame grows so that its north and south sides are of length $p(6c + 2)$ and its east side is of length $4 \cdot (6c + 2) + 2$ not including the corner tiles. The reason for these specific values will be explained shortly. Additionally, in α_0, the west side has grown to the point where it is two tiles away from colliding with the seed tile, but has not yet initiated the growth of any of it's horizontal arms. Let $\vec{\alpha}_0$ be the assembly sequence in $\mathcal{P}$ which starts with the seed tile and ends with α_0. So far, it should not be difficult to see how system $\mathcal{T}$ could simulate this assembly sequence. Let $\vec{\alpha}'_0$ be some assembly sequence in $\mathcal{T}$ which models $\vec{\alpha}_0$.

Next, we will consider a sequence of assembly sequences $\vec{\alpha}_1, \vec{\alpha}_2, \ldots, \vec{\alpha}_n$ in $\mathcal{P}$, each of which follows from the previous, wherein horizontal arms grow from the west side in a specific manner. We chose the value $n = 6c + 1$ for reasons which will become clear soon. By our choice of the east side length, the west side will be able to spawn $m = 6c + 2$ horizontal arms (each spaced 4 tiles apart) while still remaining 2 tiles away from colliding with the seed tile. We will call these arms $h_1, h_2, h_3, \ldots, h_m$ from north to south for convenience. Additionally, let $\tilde{y}$ be the y-coordinate which is exactly between the y-coordinates of arms h_{m-1} and h_m in $\mathcal{P}$. Accordingly, fix $\tilde{y}'$ to be any y-coordinate within the row of macrotile locations in $\mathcal{T}$ corresponding to the tile locations in $\mathcal{P}$ at y-coordinate $\tilde{y}$.

We define the assembly sequence $\vec{\alpha}_i$ $(i = 1, 2, \ldots, n)$ immediately following the growth of assembly sequence $\vec{\alpha}_{i-1}$ such that: (1) tiles attach so as to grow arm h_i until it collides with the east side, and (2) p tiles attach to the end of arm h_m. Note that in each assembly sequence arm h_m only grows partially by p tiles. We've chosen the width of our assembly so that even if there is an assembly sequence corresponding to

each arm north of h_m, it will not collide with the east side. Now for each assembly sequence $\vec{\alpha}_i$ in $\mathcal{P}$, let $\vec{\alpha}'_i$ be the corresponding assembly sequence in $\mathcal{T}$. Corresponding to each tile attachment in $\vec{\alpha}_i$, there may be several tile attachments in $\vec{\alpha}'_i$. During these attachments in the macrotile locations of $\mathcal{T}$, we will keep track of a specific condition, namely tiles being placed at y-coordinate $\tilde{y}'$. Including the fuzz regions surrounding the east and west arms, there are only $6c$ tile locations in $\mathcal{T}$ at y-coordinate $\tilde{y}'$ where tiles could be placed which are not too far away to invalidate the simulation. Because of our choice of $n = 6c + 1$, at least one of the assembly sequences, say $\vec{\alpha}'_j$, must occur without a tile being placed at y-coordinate $\tilde{y}'$.

Now we define the assembly sequence $\vec{\beta}$ in $\mathcal{P}$ as follows. First we follow the assembly sequences $\vec{\alpha}_0, \vec{\alpha}_1, \ldots, \vec{\alpha}_{j-1}$ in order. Then we follow assembly sequence $\vec{\alpha}_j$ up to the attachment of the second to last tile of arm h_j. We then deviate from our assembly sequences and skip the attachment of the last tile of arm h_j. Instead, we grow arm h_m, stopping one tile short of colliding with the east side of our assembly. Finally, we attach the remaining 2 tiles of the west side, colliding with the seed tile. It shouldn't be too difficult to see that $\vec{\beta}$ is a valid assembly sequence in $\mathcal{P}$. Additionally, notice that $\vec{\beta}$ is a terminal assembly in $\mathcal{P}$, since by the planarity constraint, it's now impossible for any of the arms inside the assembly to continue growth. Therefore, if we define $\vec{\gamma}$ to be the assembly sequence $\vec{\beta}$ followed by the attachment of the remaining tile in arm h_j, $\vec{\gamma}$ would not be a valid assembly sequence in $\mathcal{P}$. Despite this, we can construct an assembly sequence $\vec{\gamma}'$ in $\mathcal{T}$ which models $\vec{\gamma}$, proving that $\mathcal{T}$ does not correctly simulate $\mathcal{P}$.

To do this, we construct the assembly sequence $\vec{\gamma}'$ in $\mathcal{T}$ as follows. First, we follow the assembly sequences $\vec{\alpha}'_0, \ldots, \vec{\alpha}'_{j-1}$. Then, we follow assembly sequence $\vec{\alpha}'_j$ up to but not including the tile attachment which would cause the macrotile corresponding to the final tile of h_j to resolve. Next we continue following $\vec{\alpha}'_j$, but we skip any tile attachments north of y-coordinate $\tilde{y}'$. Because during $\vec{\alpha}'_j$ no tiles attached in this y-coordinate, this does not interfere with the growth of the macrotiles corresponding to the next p tiles in arm h_m. During these attachments south of $\tilde{y}'$, we consider a rectangular window w with dimension $p + 1 \times 3c$ and note that by our choice of p, it must be possible to position translate w along the arm in two ways, both with identical window movies. By the Window Movie Lemma, all tile attachments possible in the first of these translated windows must be possible in the second. These tile attachments can then be repeated (or "pumped") until blocked by a tile on the east side. Note that because the first pumped sequence of tile attachments didn't grow north of $\tilde{y}'$, the remaining attachments won't either. If, upon collision with the east side, the macrotile corresponding to the final tile of h_m resolves, then we reach a contradiction since we can then resolve the final macrotile corresponding to h_j with a single tile attachment after the arm h_m supposedly blocked off the region containing h_j. Otherwise, we can continue with an arbitrary sequence of tile attachments in $\mathcal{T}$ corresponding to the remaining two tiles of the west side in $\mathcal{P}$. These tile attachments cannot influence the region closed off by the collision between the macrotiles corresponding to h_m and the east wall, and therefore the attachment of a single tile in the macrotile corresponding to the final tile of h_j will still lead to the macrotile resolving, a contradiction.

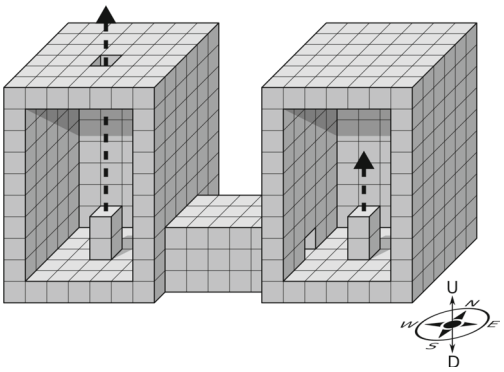

Fig. 9 Cut-away view of system $\mathcal{S}$ from the proof of Theorem 4. Two chambers are connected by a thin tunnel. Pillars growing inside the outer west chamber will eventually constrain the region within the chambers, at which point, the pillar growing in the inner east chamber will no longer be able to continue growth

6 The 3DaTAM Cannot Simulate the SaTAM

The proof of Theorem 4 is similar in principle to the proof of Theorem 3, albeit with a slightly different system which takes advantage of the differences between 2D and 3D.

Theorem 4 *There exists an SaTAM system $\mathcal{S}$ which cannot be simulated by any 3DaTAM system.*

High-level Overview

The system $\mathcal{S}$ for this result, as illustrated in Fig. 9, initially grows 2 nearly sealed chambers connected by a thin tunnel which allows for a diffusion path between them. These chambers both have a fixed base size of 9×9, but they can grow to have an arbitrary height in a way similar to the frame of the system used in the proof of Theorem 3. Once fully grown, the ceiling of one chamber contains a single tile wide opening which is the only way for tiles to diffuse into the chambers from outside; we call the chamber with this hole the *outer chamber* and the other one the *inner chamber*. Additionally, from the bottoms of both chambers, pillars can grow upwards to an arbitrary height by the attachment of copies of tiles with identical tile types. The pillar in the inner chamber will eventually crash into the ceiling *or* until the pillar in the outer chamber grows tall enough to plug the opening in its ceiling and constrain the space inside. We show that $\mathcal{S}$ cannot be simulated by any 3DaTAM system by showing that, in any potential simulating system, under the right conditions, although unwanted, it must still possible for the inner chamber pillar to continue growth even after the outer chamber pillar has sealed the chambers. To do this, we note that during some supposed simulation, the only way for the pillar in the inner chamber to "know" that the chambers have been sealed, is for tiles to attach inside of the tunnel. Consequently, because the tunnel is thin with a cross-section made of a hollow 3×3 square, the chambers can only communicate with each other a finite amount of times during a simulation. Specifically, if the scale factor of the simulation is c, then the number of tiles that can be placed in any x-coordinate corresponding to the tunnel is bounded by $5c \times 5c$ which includes any potential tiles growing in the fuzz adjacent to the macrotiles of the tunnel. Therefore, by a simple counting argument, if we initially grew our chambers

to have a sufficiently large height, then there must exist some assembly sequence where both pillars grow by any desired number of macrotiles (which we choose to be long enough to allow pumpable growth) and during which no tile is placed in the center of the tunnel. Using the Window Movie Lemma, we then construct an assembly sequence where the outer chamber pumps to constrain the chambers. Because during this assembly sequence, no tiles are placed in the center of the tunnel, there is nothing to stop the inner chamber pillar from also being pumped. Such an assembly sequence must be possible in any 3DaTAM system which supposedly simulates our system $\mathcal{S}$, and since this assembly sequence corresponds to one which is invalid in the SaTAM, such a simulation is impossible.

Proof Here we use the convention that the cardinal directions north, south, east, west, up, and down correspond to $+y$, $-y$, $+x$, $-x$, $+z$, and $-z$ respectively. When referring to dimensions of tile constructions, we use width, length, and height to refer to dimensions in the x, y, and z axis respectively.

Let $\mathcal{S}$ be the SaTAM system, illustrated in Fig. 9, described as follows. From the seed, tiles attach to form two boxes, one on the east which we will call the *inner chamber* and one on the west which we will call the *outer chamber*, connected by a thin tunnel which separates the chambers by a distance of 5. The base of each chamber is a 9×9 square of tiles and each chamber can grow to have an arbitrary height before a special tile attaches to initiate the growth of their ceilings. The ceiling of the inner chamber is solid, but the ceiling of the outer chamber has a single tile opening in its center. Inside each chamber, a pillar of tiles can grow upwards from the center of the base. These pillars are each made of copies of the same tile type which can attach on top of each other allowing the pillars to grow arbitrarily tall. The tunnel has a cross section of a hollow 3×3 square allowing for tiles to diffuse into the inner chamber until the pillar in the outer chamber has grown tall enough to plug the opening in the ceiling and constrain the region inside.

Now suppose, for contradiction, that there exists a 3DaTAM system $\mathcal{T}$ which simulates $\mathcal{S}$ using tileset T, scale factor c, and macrotile representation function R. First, we define a few constants whose values will be important during the proof. Let $p = (9c^2)!(|T|+1)^{9c^2}$. This is an upper bound on the number of orders in which tiles from T (including no tile) can be placed in a $3c \times 3c$ square. We also define $b = 25c^2$ and let $h = (p+1)(b+2)+2$ which will be used as the height of our chambers and whose value will become clear shortly. We, now consider a few assembly sequences in $\mathcal{S}$ which, by our assumption, $\mathcal{T}$ must be able to simulate. First let $\vec{\alpha}_0^{\mathcal{S}}$ be the assembly sequence in $\mathcal{S}$ during which tiles attach to the seed to complete the growth of both chambers so that both have an interior height of h (i.e. not including the base and ceiling) and both pillars grow to a height of 2. We'll refer to the last assembly in $\vec{\alpha}_0^{\mathcal{S}}$ as $\alpha_0^{\mathcal{S}}$. Additionally, we consider a series of continuations of this assembly sequence which we will define inductively; so for $k = 0, \ldots, b$, let $\vec{\alpha}_{k+1}^{\mathcal{S}}$ be the assembly sequence which begins at the assembly $\alpha_k^{\mathcal{S}}$ and during which the outer chamber pillar grows by $p+1$ tiles followed by the inner chamber pillar growing by $p+1$ tiles. To complete the inductive definition, let $\alpha_{k+1}^{\mathcal{S}}$ be the final assembly in the assembly sequence $\vec{\alpha}_{k+1}^{\mathcal{S}}$. Notice that, during the assembly sequences $\alpha_k^{\mathcal{S}}$ for $k = 1, \ldots, b+1$, each pillar will grow by a height of $p+1$ tiles which, by our definition of h to be $(p+1)(b+2)+2$,

means that neither pillar has reached the ceiling yet. We then define $\vec{\alpha}^{\mathcal{S}}$ to be the concatenation of each of these assembly sequences in order.

Because we assumed that $\mathcal{T}$ simulates $\mathcal{S}$, there must be an assembly sequence in $\mathcal{T}$ which simulates the growth of assembly sequence $\vec{\alpha}^{\mathcal{S}}$. Let $\vec{\alpha}^{\mathcal{T}}$ be such assembly sequence and, for $k = 0, \ldots, b+1$, let $\alpha_k^{\mathcal{T}}$ be the first assembly in $\vec{\alpha}^{\mathcal{T}}$ which maps under R^* to $\alpha_k^{\mathcal{S}}$. Then for convenience, we divide the assembly sequence $\vec{\alpha}^{\mathcal{T}}$ into subsequences $\vec{\alpha}_k^{\mathcal{T}}$ $(k = 0, \ldots, 5c+1)$ so that $\vec{\alpha}_k^{\mathcal{T}}$ simulates the assembly sequence $\vec{\alpha}_k^{\mathcal{S}}$ and ends with the assembly $\alpha_k^{\mathcal{T}}$. We will now use these assembly sequences in $\mathcal{T}$ to construct a new assembly sequence in $\mathcal{T}$ which cannot possibly correspond to a valid assembly sequence in $\mathcal{S}$. First, let x_t be the x-coordinate of the center of the tunnel in $calT$. For the macrotiles in the inner chamber to "know" anything about the macro tiles in the outer chamber, tiles must attach in a location with x-coordinate x_t. In $\mathcal{T}$, a cross section of the tunnel at x-coordinate x_t is a 3×3 macrotile square and, if we include fuzz, this means that no tile in $\mathcal{T}$ will be able to attach outside of the $5c \times 5c$ square surrounding the tunnel at x-coordinate x_t. Consequently, at most $25c^2$ (our choice of value for b) tiles will be able to attach in locations with x-coordinate x_t during the assembly sequence $\vec{\alpha}^{\mathcal{T}}$. Therefore, there must exist some index j $(1 \leq j \leq b+1)$, such that during the assembly sequence $\vec{\alpha}_j^{\mathcal{T}}$, no tile is ever placed at x-coordinate x_t.

Now notice that during assembly sequence $\vec{\alpha}_j^{\mathcal{T}}$, tiles attach to grow both pillars by a height of $p+1$. Our definition of p will allow us to use the Window Movie Lemma as follows. First, let w be the window defined as the boundary of a box with x and y dimensions $3c$ and with z dimension $p+2$. Then note that because each pillar started at a height of 2 macrotiles before any of the assembly sequences $\vec{\alpha}_1^{\mathcal{T}}, \ldots, \vec{\alpha}_{b+1}^{\mathcal{T}}$, none of the fuzz adjacent to any macrotiles in $\vec{\alpha}_j^{\mathcal{T}}$ will be adjacent to any macrotiles except those which resolve in $\vec{\alpha}_j^{\mathcal{T}}$. Because p was defined as an upper bound on the number of orders in which tiles from T can attach in a $3c \times 3c$ square and, since we are growing $p+1$ macrotiles on each pillar during $\vec{\alpha}_j^{\mathcal{T}}$, there must be two ways to translate our window w, say w_1^{out} and w_2^{out}, so that it is centered on the outer chamber pillar, such that w_1^{out} and w_2^{out} have identical window movies during the assembly sequence $\vec{\alpha}_j^{\mathcal{T}}$. By the Window Movie Lemma, we can therefore create a new assembly sequence $\vec{\beta}^{\text{out}}$ which begins the same as $\vec{\alpha}_j^{\mathcal{T}}$ but during which all of the tile placements in the region bounded by w_1^{out} also occur in the region bounded by w_2^{out}. We can do the same for the pillar in the inner chamber to define an assembly sequence $\vec{\beta}^{\text{in}}$ analogously. We can then repeat this process to "pump" both pillars upwards by repeating the tile attachments occurring in the respective window regions, noting again that no tile is ever placed in x-coordinate x_t so neither pillar's growth can affect the other.

We can therefore construct an assembly sequence $\vec{\gamma}$ in the following way. First we continue to pump the growth of the outer chamber pillar until the final macrotile resolves so that the pillar has grown to height h. We do have to be a bit careful though, because during the final pumping iteration, the assumptions of the Window Movie Lemma will no longer hold. This is because when we get close enough to the opening in the ceiling, the macrotiles of the ceiling and surrounding fuzz will alter the window movie. If we carefully consider the growth that occurs during the pumping however, this is not a problem. To see why, recall that growth is not allowed to occur in any

locations which are not in the fuzz adjacent to resolved macrotiles. Consequently, in the prior pumping iterations, as tiles attach in a macrotile location, say m, on top of the pillar, no tiles have been able to attach in those macrotile locations adjacent to m before it resolves. Consequently, even though the Window Movie Lemma no longer holds, we can still repeat the same sequence of tile attachments up to the point where the macrotile resolves to form a height h pillar, as none of the tiles in the ceiling fuzz occupy locations which would be occupied by the pillar and could not prevent the necessary attachments. After growing the outer chamber pillar to a height of h macrotiles, we continue the assembly sequence $\vec{\gamma}$ by performing the same tile attachments that occurred in $\vec{\beta}^{\text{in}}$, corresponding to tiles attaching to the inner chamber pillar. We then let the assembly sequence $\vec{\delta}$ be the assembly sequence formed by concatenating the assembly sequences $\vec{\alpha}_0^{\mathcal{T}}, \ldots, \vec{\alpha}_{j-1}$ and the assembly sequence $\vec{\gamma}$. This assembly sequence is valid in $\mathcal{T}$, but under R it maps to an assembly sequence in $\mathcal{S}$ wherein the inner chamber pillar continues growth after the outer chamber pillar has grown tall enough to constrain the region. This is a contradiction and since we made no prior assumptions about $\mathcal{T}$, it cannot be the case that $\mathcal{T}$ correctly simulates $\mathcal{S}$. □

7 The PaTAM can Simulate the Directed PaTAM

Theorem 5 *There exists a universal Planar aTAM tile set S that can simulate any directed PaTAM system.*

High-level overview

Despite the fact that both the PaTAM and directed PaTAM are not intrinsically universal for themselves [25], using tools from [23, 25] we are able to construct a PaTAM tile set capable of simulating arbitrary directed PaTAM systems. Here we outline the process by which a PaTAM tileset S can simulate any given directed PaTAM system $\mathcal{T}$. The tileset S is universal, meaning that regardless of the directed PaTAM system $\mathcal{T}$, the same tileset will be used at a fixed binding threshold, with only the seed of the simulating system changing to accommodate $\mathcal{T}$.

Given a directed PaTAM system $\mathcal{T}$, we define a simulating system $\mathcal{S}$ using a fixed tile set at binding threshold 2. The seed of $\mathcal{S}$ consists of already-resolved macrotiles in the same configuration as the seed of $\mathcal{T}$. Each macrotile in $\mathcal{S}$ consists of a 9×9 grid of structures we call *component blocks* (CBs) which are each made of many smaller tile-based constructions and which each store an encoding of the system $\mathcal{T}$ along with a bit of extra data in the form of specific glues on some of its tiles. The CBs of a macrotile each perform calculations using tiles which emulate Turing machines to determine how they should grow and whether or not the macrotile can resolve given the current information regarding the surrounding macrotiles.

Each CB essentially behaves like an individual tile on the 9×9 grid and we can think of CBs as growing in one of two ways. Either the CB grows using tile attachments from another adjacent CB in a way analogous to a τ-strength tile attachment, or a CB can grow in the gap between two adjacent CBs in certain locations of the grid designated as *probe regions*. This is analogous to a tile attachment that occurs by cooperative binding

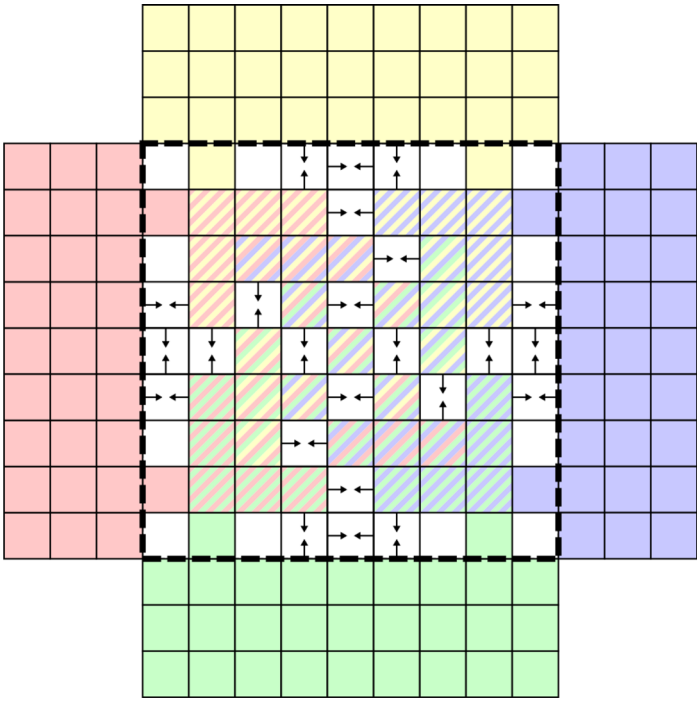

Fig. 10 A schematic describing the 9×9 grid of potential component blocks which may appear in a macrotile location. Squares containing two arrows indicate a grid location which may contain a probe region. Parts of adjacent macrotiles are indicated by the colored squares surrounding the macrotile. The colors of the grid locations within a macrotile describe which of the adjacent macrotiles each component block may have information from (Color figure online)

between two opposing tiles (which we refer to as *across-the-gap* cooperation). These "cooperative attachments" between CBs are used to consolidate information between the CBs. For instance, one CB might contain information encoded about the north adjacent macrotile and one might contain information about the west; in the probe region between them, a new CB can grow which will contain the information about both which it can then use to determine if a tile attachment in $\mathcal{T}$ would be possible in the tile location corresponding to the macrotile. Figure 10 illustrates the layout of a macrotile into CB locations with these probe regions indicated by squares with two opposing arrows.

Probe regions are CB locations in which two adjacent CBs, on opposite sides, can present structures called *probes* which are long, thin structures that grow from the surrounding CBs towards the center of a CB location. Each probe that grows in a probe region, indicates some possible combination of information from surrounding macrotiles and grows in a unique position according to this information. The length of a probe is chosen to be just shy of the center of the CB location, so that when two probes align from opposing sides of the probe region, there will be exactly a single tile wide gap between them. This gap allows a tile to cooperatively attach and grow along the sides of the probes to recover the information from both. Otherwise, if no probes

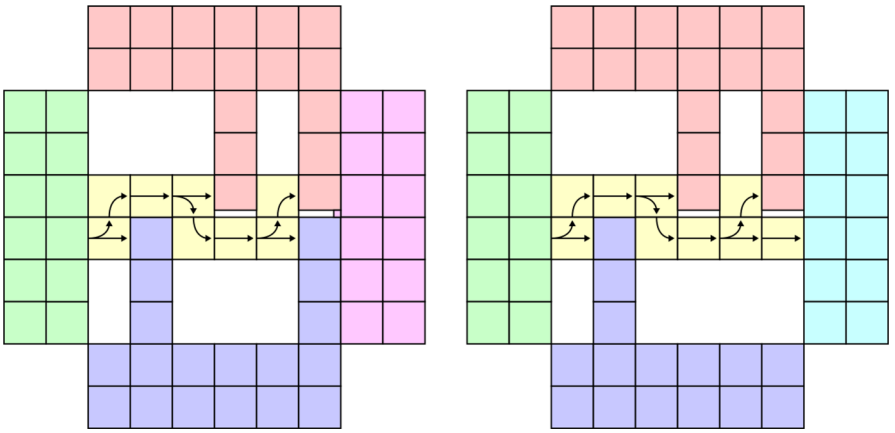

Fig. 11 Probe regions between component blocks. The red and blue CBs present probes made from TM blocks into the grid location between them. The yellow TM blocks are spawned from those in the green CB and can only pass through the probe region if none of the probes align. In the case that the probes do align (left) the yellow TM blocks will not be able to pass and the CB on the right is initiated by cooperative growth between the probes. If the probes do not align (right), then the yellow TM blocks can pass through and initiate growth of the right CB. Therefore depending on the alignment of the probes, the CB to the right of the probe region is determined by either the CBs from which the probes grew or the CB on the left of the probe region (Color figure online)

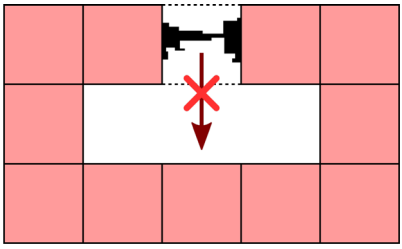

Fig. 12 When checking for across-the-gap cooperation during a simulation, tiles can't naively span the entire gap without disconnecting two regions of space

in a probe region align, there will be enough room for the components that make up a CB to squeeze in between the probes from one side of the probe region to another. Figure 11 illustrates two scenarios involving probe regions.

Probe regions were introduced in [23] to solve the problem illustrated in Fig. 12. Naively when simulating a tile system, to check for macrotiles which may cooperate across-the-gap, tiles must grow to query both adjacent macrotiles and determine if the attachment is possible. This however necessarily separates regions of space and in the case of planar systems also constrains one before it has been determined if the attachment can even occur. If it cannot, then tiles will no longer be able to attach in the constrained region and the simulation will likely end up being invalid. Probes avoid this problem by aligning exactly when across-the-gap cooperation is possible while still allowing tile structures to grow through if they don't align.

Now that we have an idea of how the component blocks and probe regions behave we describe the protocol for resolving a macrotile by highlighting a few important cases. Growth within a macrotile begins when one or more of the surrounding macrotiles

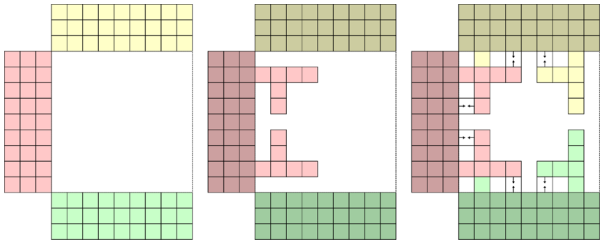

Fig. 13 Hands made of component blocks growing from surrounding macrotiles

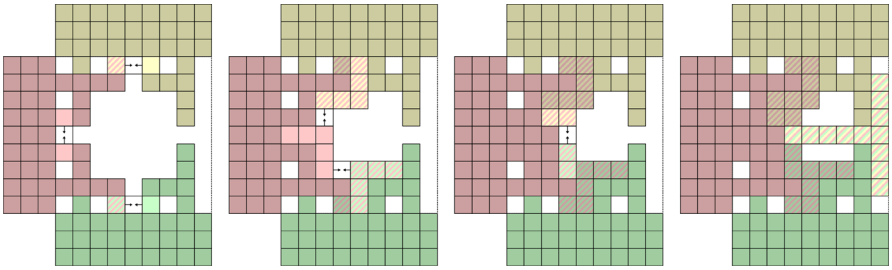

Fig. 14 Once the hands have grown, CBs cooperate until information from all sides has been combined into a single CB. Then the macrotile can resolve

resolve and tiles begin to attach within the macrotile. From a surrounding macrotile, the protocol always begins by the growth of two "T"-shaped structures made from CBs called *hands* illustrated in Fig. 13. Note that two adjacent surrounding macrotiles may both attempt to grow hands in the same location. This is handled by a single point of competition and the first surrounding macrotile for placing a tile in the closest corner of the shared hand locations is allowed to place theirs. Between the hands and the surrounding macrotiles probes are grown in the regions indicated on the right of Fig. 13 which allows a CB to "attach" cooperatively to combine information from both the hand and nearby macrotile. In some cases this information may be redundant, but with two or more surrounding macrotiles at least one location will always be able to combine information from two macrotiles (Fig. 14).

The CBs resulting from cooperation between the hands and surrounding macrotiles then cooperate once again and CBs grow along the hands to form clockwise elbows with additional probe regions between them. CBs then cooperatively attach between these elbows and cooperate again near the center of the tile to eventually combine all of the information from the surrounding macrotiles. Once this occurs, the CB which "attaches" in the center of the tile contains the information from all sides. If the surrounding macrotiles represent tiles in $\mathcal{T}$ capable of placing a tile, additional CBs can grow to the remaining sides to present this information to the remaining sides and repeat the procedure in the adjacent macrotile locations.

In the case that an across-the-gap cooperation is possible in $\mathcal{T}$, the protocol deviates slightly. Illustrated in Fig. 15, if across-the-gap cooperation is possible between the east and west macrotiles, their hands will share a probe region with aligned probes. Consequently, a CB can grow in that location and resolve the macrotile. This growth

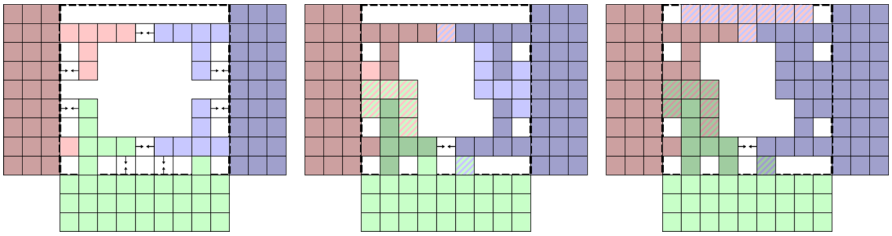

Fig. 15 Probe regions between opposing macrotiles can check for across-the-gap cooperation

may constrain the region to the south, halting any tile attachments and CB growth in the south side of the macrotile, but this doesn't matter since the macrotile will only need to start the process in adjacent macrotiles that haven't yet resolved. The described protocol is robust to different orders of hand growth and different numbers of surrounding macrotiles, including those that don't end up contributing to macrotile resolution. If at any time a CB has sufficient information to determine how the macrotile should resolve, it begins growth to the center and then surrounding edges of the macrotile. This process will not be interrupted by other CBs since we are simulating a directed system where at most one unique tile can attach in each location.

Proof Our proof of Theorem 5 is by construction. Therefore, we now give a full description of the components of that construction and describe how it works. □

7.1 Tile Gadgets

This construction makes use of a few *tile gadgets*, components which perform some specific task or grow in a specific pattern, which have been adapted from the existing tile-assembly literature. Much like how algorithms are often described using pseudocode using existing data-structures, we will describe our construction in terms of these gadgets.

Turing Machine Blocks
Used in [17] to assemble finite shapes with asymptotically optimal tile-complexity, a Turing machine block is a tile gadget which simulates space and time bounded Turing machines within a square region of space. Each side of a TM block is designated either as an input or an output. Both the input and outputs of a TM block are encoded by the glues presented along a row of tiles. These glues can be used to encode data in the form of bits if special glues are designated to act as 0 s and 1 s, and additionally, the glues presented by the output sides can be used to interface with other tile gadgets or initiate the growth of an adjacent TM block. Consequently, TM blocks can grow to form arbitrary computable patterns.

Probes and Probe Regions
Initially defined in [23], probes are simply tall rows of tiles placed at specific locations of a macrotile. Probes solve the problem illustrated in Fig. 12 of trying to simulate across-the-gap (AtG) cooperation. Naively, it would seem that it's necessary to block-off a region of space to learn if two opposite macrotiles could cooperate to place a tile

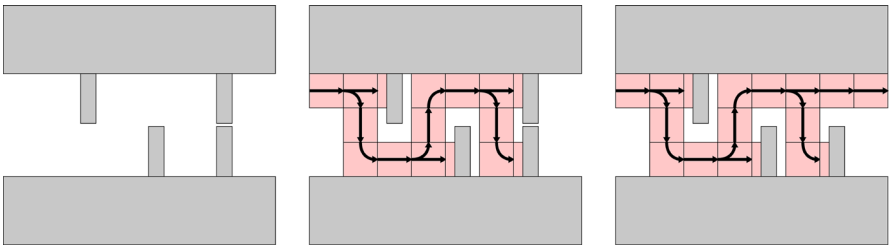

Fig. 16 An illustration of probe regions. In the case that the probes do not meet, it will be possible for a TM blocks to grow around the probes. Otherwise, probes will block the growth of TM blocks and across-the-gap cooperation between the probes can initiate the growth of different TM blocks which perform a different task

in your location. We can avoid this by specifying regions where sequences of probes grow into locations corresponding to pairs of tiles which could potentially cooperate AtG. If AtG cooperation isn't possible between the relevant macrotiles, then the probes will be spaced sufficiently far to allow a data path to navigate through the probe region; otherwise, two probes will meet, with a gap of a single tile between them which is insufficient for a data path to fit. Instead a tile can cooperatively attach between the probes and initiate the growth of a new data path indicating that the current macrotile can resolve into the tile placed by the AtG cooperation (Fig. 16).

7.2 Macrotile Protocol

Given a DPaTAM system $\mathcal{T}$, we describe the simulation of $\mathcal{T}$ by some PaTAM system $\mathcal{S}$ by describing the process by which tiles attach within macrotiles of $\mathcal{S}$ to resolve into a tile in $\mathcal{T}$. Growth within a macrotile location begins once at least one adjacent macrotile has resolved and our macrotiles must be robust to the arbitrary orders in which adjacent macrotiles can resolve. In this way, tile attachments within a macrotile behave similarly to a concurrent algorithm with special care needed to handle race conditions and avoid deadlocks when undesirable.

Macrotiles and Component Blocks

The fundamental component of used in our protocol are tile gadgets which we call *component blocks* (CBs) which are made from several TM blocks and which may potentially initiate the growth of probe regions. Each macrotile in $\mathcal{S}$ will consist of a 9x9 grid of CBs and in turn, each CB consists of a $2k \times 2k$ grid of TM blocks where k is the number of ways of choosing 4 tile glues from $\mathcal{T}$. That is $k = g^4$ where g is the number of glues appearing on tile types from $\mathcal{T}$. Each of the TM blocks within a CB is large enough to store the following information:

- a description of the simulated system $\mathcal{T}$,
- a sequence of upto 4 glues from $\mathcal{T}$,
- its position within the CB,
- the CBs position within a macrotile, and
- a description of the macrotile algorithm protocol which acts as the algorithm to be performed.

We will refer back to this list later to determine the scale factor of the simulation once the protocol has been described.

Figure 10 illustrates the 9×9 grid of component blocks including those which may contain probe regions. Each probe region may contain probes which grow from the adjacent CBs indicated by the directions of the arrows. The probes in a probe region are made from rows of TM blocks spanning half of the width of a CB though, one of the CBs which spawns a probe is selected beforehand to grow just shy of half-way through the probe region. This creates a single-tile wide gap between two aligned probes in which tiles can cooperatively attach to initiate the growth of other TM blocks which depend on the alignment of probes. Figure 11 illustrates two scenarios: one where two probes align in a probe region, in which case no path of TM blocks can pass through the region and in which cooperation between the probes determines the next CB, and a scenario where no probes align, in which case a path of TM blocks can pass to determine the next CB.

We can think of the component blocks as behaving similarly to individual tiles, with probe regions facilitating across-the-gap cooperation, and which perform the algorithm of simulating $\mathcal{T}$ in the 9×9 grid constituting a macrotile. These "tiles" grow along a path accumulating information regarding the surrounding macrotiles until they reach the center of the macrotile. Once there, by design of our protocol, the TM blocks making up the CB will have enough information to determine the tile from $\mathcal{T}$ into which the macrotile should resolve. The specified probe regions indicate potential locations in which two CBs can "consolidate" information about the adjacent macrotiles while also allowing TM blocks to pass through in the case that probes don't align. Consequently, the size of the CBs, being $2k \times 2k$ TM blocks, was chosen so that there are always enough probe locations for every combination of glues presented by the adjacent macrotiles. The additional factor of 2 means that each probe has enough space for TM blocks to squeeze around even adjacent probes.

Protocol for One Surrounding Macrotile

We will describe the protocol which occurs in the 9×9 grid of component blocks as though each CB is a tile. Cooperative binding of these "tiles" always occurs in a designated probe region and is facilitated by the probes, while τ-strength attachments are facilitated by the TM blocks which make up each CB "tile". In other words, τ-strength attachments of CBs is simply a fixed pattern of TM block growth which cause the CBs to grow along some predetermined path which may depend on the information stored in the TM blocks.

Illustrated in Fig. 17, from an adjacent macrotile, CBs first grow into two mirrored "T"-shaped structures called *hands*. In the probe regions between these hands and the adjacent macrotile, CBs then grow cooperatively. Because, for the time being, we are only dealing with a single adjacent macrotile, these cooperative "attachments" just cause CBs to grow into the center of the first row of the macrotile. Then one of two things may happen. If the adjacent macrotile corresponds to a tile in $\mathcal{T}$ whose glue can allow a τ-strength attachment, then the TM blocks which make up the CBs has enough information to resolve the current macrotile. In this case, τ-strength attachments cause the CBs to grow into the center of the macrotile and eventually cause the macrotile to resolve, as will be described later. Otherwise, a path of CBs will simply grow along

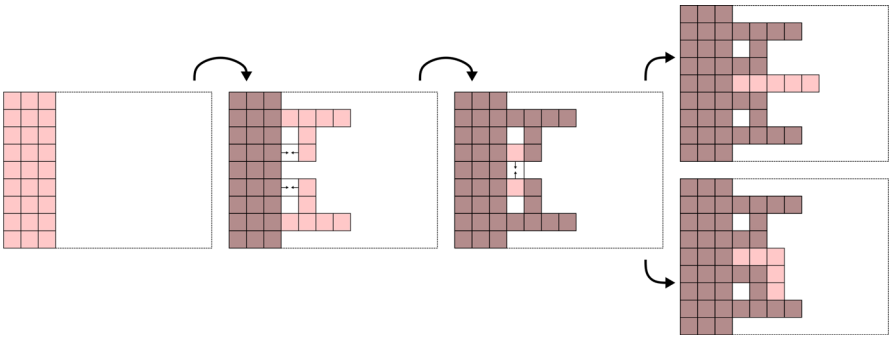

Fig. 17 The pattern of CB growth from an adjacent macrotile. Starting from the adjacent macrotile, CBs will grow into mirrored *hands* which cooperate to allow CBs to attach in the center of the first row in the macrotile. If this macrotile corresponds to a τ-strength attachment in $\mathcal{T}$, then a row of CBs will attach to the center (top) and will eventually resolve the macrotile, otherwise, they grow along a clockwise turn (bottom) to await growth from adjacent macrotiles

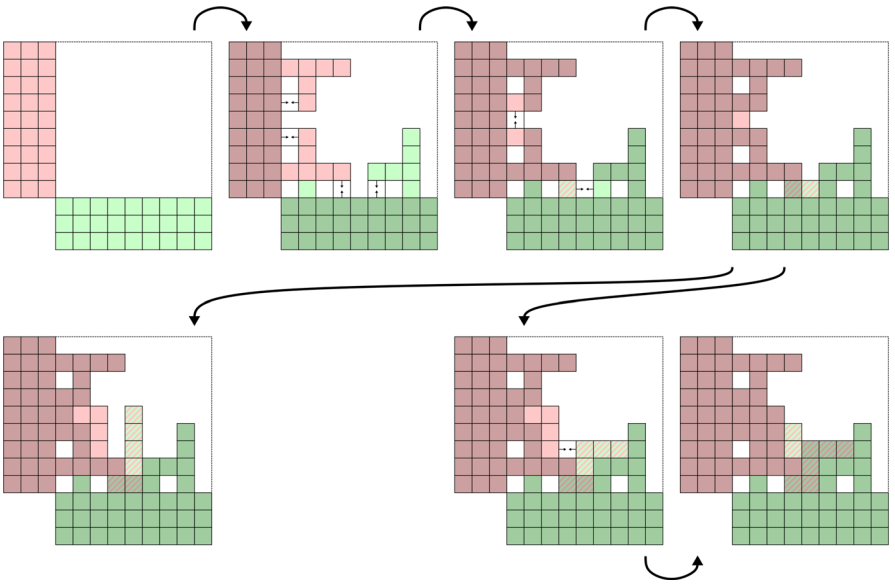

Fig. 18 With two adjacent surrounding macrotiles, the protocol occurs similarly to before, but the two surround macrotiles will compete to place one of their hands

the hand on the clockwise side of the macrotile and no further growth will occur until another adjacent macrotile resolves.

Protocol for Two Adjacent Surrounding Macrotiles

Figure 18 illustrates the protocol for two surrounding macrotiles which appear on adjacent sides of the current macrotile. We note that the surrounding macrotiles may resolve at different times and therefore our protocol is designed to handle different orders of CB attachment. Growth begins in much the same way as the case above with both surrounding macrotiles attempting to first grow their hands. Notice though that

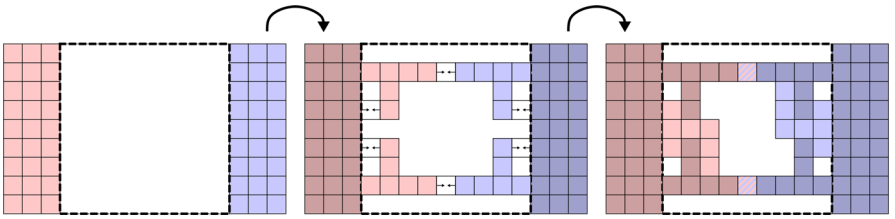

Fig. 19 In the case of two surrounding macrotiles which meet across-the-gap, probe regions on the end of the respective hands cooperate, if possible

each will attempt to grow one of their hands in the same location. To handle this, we simply require that during the growth of the hands, the TM blocks which make up the CB blocks shared by both hands start by placing a single tile in the shared corner. Whichever hand is able to place that tile first is the one which grows its hand. In Fig. 18 this happens to be the hand from the macrotile to the west, but it could have just as well been the one from the south.

Once the hands are grown, cooperative "attachment" of CBs occurs between the hands and the surrounding macrotiles just like before, but in this case one of the cooperative attachments will happen between one of the macrotiles and a hand which originated from the other macrotile. This cooperative CB attachment will therefore be able to consolidate the information from both adjacent macrotiles, storing both glues in the TM blocks which make the CB. If this information is enough to allow the macrotile to resolve, that is if the surrounding macrotiles represent tiles in $\mathcal{T}$ which can cooperate to allow the attachment of a tile in this location, then CBs will "attach" with τ-strength to the center of the macrotile which will eventually cause it to resolve. Otherwise, they both grow to form clockwise elbows just as in the case with only 1 surrounding macrotile. This time however, a probe region exists between the two adjacent elbows which will allow the further attachment of a few CBs. These attachments will not do anything until another surrounding macrotile resolves since no tile in $\mathcal{T}$ can yet attach in the location corresponding to this macrotile.

Note that this process is robust to the order in which surrounding macrotiles resolves. Even if the west macrotile resolves well before the south and grows according to the protocol for a single surrounding macrotile, this protocol is still able to occur with the south macrotile being the one which loses the competition to place its hand. The situation is symmetrical so that it occurs similarly even if the south macrotile wins the competition. Additionally, even if the first surrounding macrotile to arrive corresponds to a tile in $\mathcal{T}$ which is able to form a τ-strength attachment in the current macrotile, the attachments formed by the other macrotile will not interfere.

Protocol for Two Opposite Surrounding Macrotiles

In the case that the two surrounding macrotiles are opposite each other, the probe regions between the hands grown by both will facilitate the resolution of the macrotile as illustrated in Fig. 19. Once the hands are grown, if the two opposing macrotiles correspond to tiles in $\mathcal{T}$ capable of across-the-gap cooperation, then the probe regions between the opposing hands will have aligned probes. Using these probes, the TM blocks which make up the CB can consolidate the information from both surrounding

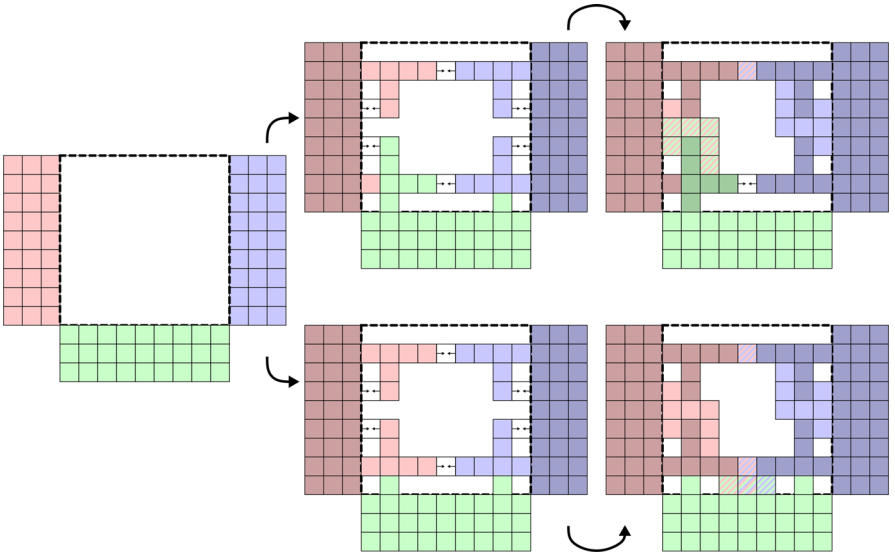

Fig. 20 In the case that two surrounding macrotiles must simulate an across-the-gap cooperative attachment, a third surrounding macrotile may attempt to cooperate with either, but cannot prevent the cooperation on both sides of the current macrotile

macrotiles and consequently are able to determine the tile into which the current macrotile will resolve. Note that this case is slightly different from the previous cases since CBs will not grow into the center of the macrotile. In fact, the interior of the macrotile will become a constrained region into which no additional tiles can diffuse once both have grown. This doesn't matter however since both of the CBs between the hands have sufficient information to determine how the macrotile should resolve and all that needs to be done is have that information propagated to the empty sides of the tiles.

Even in the case that there is another surrounding macrotile between the opposing macrotiles, the protocol allows for the successful resolution of the macrotile as illustrated in Fig. 20. There are several possibilities regarding which macrotiles win the competition to place their hands, but these will not affect the probe region on the side which does not contain a surrounding macrotile. Consequently regardless of the tile attachments that occur on the south of Fig. 20, CBs can still attach on the north side of the macrotile indicating that it resolved into the tile corresponding to an across-the-gap cooperative attachment. Any growth below the north probe region between opposing hands will not be able to affect this resolution since it will be blocked by the probes and growth along the south side is irrelevant since the macrotile does not need to indicate its resolution in that direction as the surrounding macrotile to the south has already resolved. Also, since we are simulating a directed system, the current macrotile could not resolve into any other tile anyway.

If on the other hand, the two opposing macrotiles do not represent tiles in $\mathcal{T}$ capable of across-the-gap cooperation, then no probes in the region shared between opposing hands will align and TM blocks will be able to grow in between, allowing the protocol

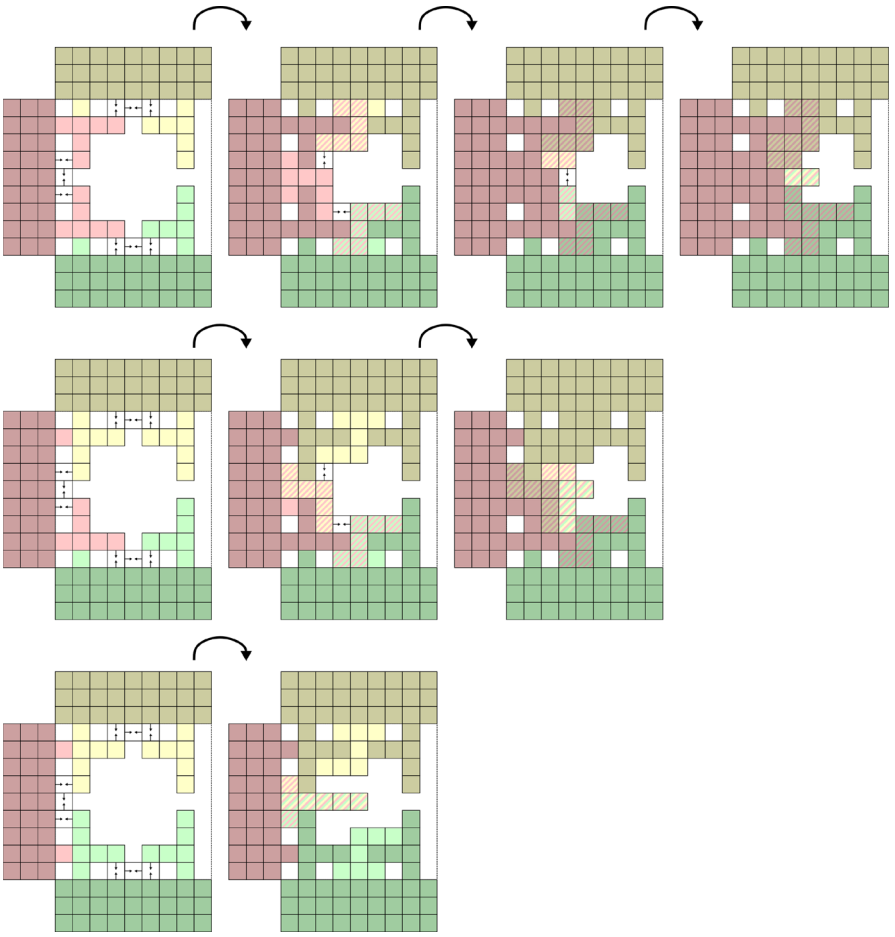

Fig. 21 With 3 incoming macrotiles, up to symmetry, there are 3 fundamentally different ways that hands can grow. In all cases, it is possible for information from all 3 macrotiles to be collected in the center

to continue to other cases. In other words, opposing macrotiles which aren't capable of across-the-gap cooperation will not interfere with the continuation of the protocol. If an additional macrotile resolves to either the north or south and represents a tile in $\mathcal{T}$ which can cooperate with either the east, west, or both surrounding macrotiles, it will still be able to do so.

Protocol for Three Surrounding Macrotiles

In the case of three surrounding macrotiles which cooperate to resolve the macrotile, there are a few cases which need to be considered regarding which surrounding macrotiles win the race to place their hands. These cases are illustrated in Fig. 21. The most complicated case occurs when the central surrounding macrotile wins both races and places both of its hands, though all cases are essentially the same. In this case, like before, all three sides grow their hands and corresponding clockwise elbows.

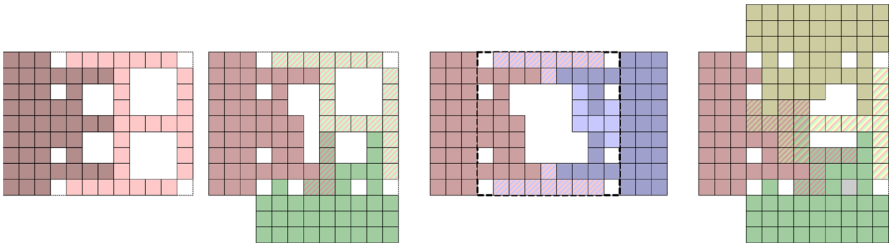

Fig. 22 Regardless of the number of surrounding macrotiles and their relative orientations, once the the CBs are able to determine how to resolve, additional CBs will grow from the center (or probe regions between across-the-gap macrotiles) to the sides which do not yet have a macrotile present. This is facillitated by the planar restriction which prevents CBs from growing towards sides with existing macrotiles

These elbows then cooperate via probes to consolidate their information with CBs from the adjacent elbows. We are then left with CBs which can again cooperate to consolidate the information from all three sides and grow into the center of the macrotile to eventually resolve. The other cases are simpler in that the CBs consolidate the information of all three surrounding macrotiles earlier. In these cases, the CBs have enough information to "know" how the macrotile should resolve and are able to grow into the center earlier.

Four Surrounding Macrotiles

In our directed PaTAM system $\mathcal{T}$, it is impossible for a tile to be placed in a location surrounded by tiles on all sides. Any tile location surrounded on all 4 sides is constrained; if all 4 tiles were required for the tile placement, it could not occur because of the diffusion restriction. If on the other hand, fewer than 4 tiles were required but it were still possible for 4 tiles to surround a location before the tile attached, the system would not be directed; the tile placement may or may not happen before the region is constrained. Such situations will never happen in our simulating system $\mathcal{S}$ because they violate the assumption that $\mathcal{T}$ is a DPaTAM system.

Resolving a Macrotile

Once a CB reaches the center of a macrotile (or when across-the-gap cooperation happens in the probe regions between opposing macrotile hands), the macrotile is ready to resolve. Upon placement of the first tile in this center CB location, the macrotile representation function R no longer maps the macrotile to empty space and instead maps to the correct tile in $\mathcal{T}$ which has been determined by the CBs from the information from the necessary surrounding macrotiles. Once this happens, additional CBs begin "attaching" to grow in each of the 4 cardinal directions in order to grow a row of tiles to each necessary side of the macrotile as illustrated in Fig. 22. These rows of tiles then act to repeat the entire process in adjacent macrotile locations, growing hands and following the described protocol.

Note that it may be the case that CBs cannot fully grow in one of the directions. This could happen if one of the surrounding macrotiles resolves after much of the process has completed in the current macrotile for instance or if one of the directions requires tile placements in a constrained region. This isn't a problem however because only those surrounding macrotiles which could correspond to locations in $\mathcal{T}$ where

valid tile attachments can occur need to "know" about the current macrotiles resolved state.

7.3 Scale Factor of the Simulation

Recall that each TM block which makes up a component block contains the following information:

- a description of the simulated system $\mathcal{T}$,
- a sequence of upto 4 glues from $\mathcal{T}$,
- its position within the CB,
- the CBs position within a macrotile, and
- a description of the macrotile algorithm protocol which acts as the algorithm to be performed.

For a given system $\mathcal{T} = (T, \sigma, \tau)$, we note that only the tileset T and binding threshold τ are necessary for the TM blocks and component blocks to determine how a macrotile should resolve. The seed σ of $\mathcal{T}$ is handled by the seed of $\mathcal{S}$ since our simulation can begin with a seed assembly made of pre-resolved macrotiles already placed in the correct configuration to represent σ.

To describe a tileset T, we note that the number of glues appearing in T is at most $4|T|$. Each glue can therefore be encoded efficiently using $O(\log(|T|))$ bits and consequently the entire tileset can be encoded using $O(|T|\log(|T|))$ bits. The binding threshold is just a positive integer and thus can be encoded using $O(\log(\tau))$ bits. For the purposes of our TM blocks, a table describing the tileset T along with the binding threshold τ can thus be encoded using $O(|T|\log(|T|) + \log(\tau))$ bits. Additionally, a sequence of 4 glues simply requires $O(\log(|T|))$ bits to encode.

The position of a TM block within a component block has coordinates between 0 and $2k$ where k is, again, the number of possible sequences of 4 glues. $k = g^4$ where g is the number of glues so $\log(2k) = \log(2g^4) = O(\log(g)) = O(\log(|T|))$ and thus the position of a TM block can be encoded using $O(\log(|T|))$ bits. Since the position of of a CB within a macrotile is always a point in $\{0, 1, \ldots, 8\}^2$ it simply requires a constant number of bits to store. Similarly, the algorithm performed by a TM block is fixed so it also requires only a constant number of bits to store. Therefore, the information stored by a TM block in our construction requires only $O(|T|\log(|T|))$ bits to encode.

Each TM block within a macrotile performs a fixed algorithm, implicitly described above. For the most part, this algorithm simply requires the TM blocks to initiate the growth of other TM blocks according to a fixed pattern to grow the current component block, but occasionally also requires querying the tileset T to determine if the known glues are sufficient to induce a tile attachment in $\mathcal{T}$. To grow in the fixed pattern, requires space and time on the order of the size of the CB which is $O(|T|^4)$. To determine if the known glues are sufficient for an attachment in $\mathcal{T}$, we can modify our naive encoding of the tileset T and binding threshold $\mathcal{T}$ with a slightly more complex encoding which consists of a table with entries for each combination of 4 glues. These entries simply store a reference to the tile type which may attach given the surrounding glues. Since we are simulating a directed system, we know that it should

never be possible for multiple tiles to attach in a single location so no entry needs to be stored with more than one possible tile. Such an encoding requires g^4 entries each of size $O(\log(|T|))$ since each holds just a single tile type's description. This table thus requires $O(|T|^4 \log(|T|))$ space and can be queried in the same amount of time.

Overall, each TM block needs to therefore be $O(|T|^4 \log(|T|))$ tiles large and since each CB is $2k = 2g^4$ TM blocks in each dimension, a component block must be $O(|T|^8 \log(|T|))$ tiles wide in the worst case. Since there are a fixed number of CBs per macrotile, this is therefore the scale factor of the simulation. While the scale factor is polynomial in size, we note that it can be significantly improved by storing the tileset look-up table separately from the TM blocks, most of which do not need to query the table. In previous iterations of our construction this was done but required a significantly more complex protocol that was much more difficult to explain and understand, so here we've presented a construction with a larger scale factor but which is easier to describe. Regardless, the purpose of this construction is to demonstrate the existence of a universal tileset and show that the PaTAM is capable of all of the dynamics of its directed subset, which we have done. Thus, Theorem 5 is proven.

8 Conclusion

Through the results in this paper we have increased the understanding of how the aspects of dimensionality, diffusion, and directedness impact the powers of tile-based self-assembling systems. We have shown many instances in which systems in one model are unable to match the behaviors of those in another model and instances in which all systems of one model can be faithfully simulated by systems in another, sometimes leading to the powers of one model being a strict subset of those of another, and sometimes to models having mutually exclusion abilities. While most of these results require highly theoretical constructions that are unlikely to be implementable via current approaches to building such systems, e.g. with DNA-based components, our goal was to provide a high-level mathematical understanding of the dynamics of self-assembly.

Clearly, as indicated by the holes in Table 1, several open questions remain. The majority of these relate to the abilities of systems in other models to simulate all systems in the Planar aTAM and Spatial aTAM that are directed. While we have dedicated much effort to answering these questions, the answers remain elusive. To summarize, we conjecture that the systems of the PaTAM and SaTAM cannot be simulated by systems in models that are not constrained by diffusion. This is because systems unconstrained by the requirement of tiles to be able to diffuse to frontier locations would have to somehow "know" which locations are contained within closed-off regions in advance so that neighboring tiles do not expose glues that would allow tiles to attach there. However, whether a region is constrained or not is a global property of an assembly, potentially influenced by arbitrarily many tile attachments at arbitrarily far away locations, and this intuitively seems impossible to coordinate in systems where the diffusion constraint isn't globally enforced by model dynamics. Despite the seemingly obvious (at least to the authors) impossibility of using systems unconstrained by diffusion to simulate those that are, the additional constraint of directedness creates

another level of difficulty. Since the systems to be simulated must be directed, it must be the case that any region which may be sealed off *always* receives the exact same tiles before the region is sealed. This implies a level of determinism that could potentially allow the creator of the simulating system to know in advance which tiles must be placed in which locations and somehow encode that information in the system itself, circumventing the limitation in dynamics to impose correctness. Therefore, any proof of impossibility would perhaps need to utilize obfuscation of such information (similar, perhaps, to techniques used in [24]), and our conjecture is that such an approach may be possible. However, perhaps we are incorrect and it is always possible to utilize that determinism to design simulating systems.

We hope that the current set of results and the techniques introduced will be useful for the development of additional theoretical and experimental self-assembly results. We are also excited to see if the remaining open questions from Table 1 can be solved, and not only what the answers may be but what new techniques may be involved.

Acknowledgements This work was supported in part by NSF Grant CAREER-1553166

Author Contributions D.H wrote the main manuscript, except for Theorems 1 and 2 and the conclusion, and prepared all figures except for 4 and 5. M.J.P wrote the statements and proofs for Theorems 1 and 2 as well as the conclusion, and prepared Figs. 4 and 5. Both authors reviewed the manuscript.

Declarations

Conflict of interest The authors declare that there were no conflicts of interest during the writing and publication of these results.

References

1. Douglas, S.M., Dietz, H., Liedl, T., Högberg, B., Graf, F., Shih, W.M.: Self-assembly of DNA into nanoscale three-dimensional shapes. Nature **459**, 414 (2009)
2. Ke, Y., Ong, L.L., Shih, W.M., Yin, P.: Three-dimensional structures self-assembled from DNA bricks. Science **338**(6111), 1177–1183 (2012)
3. Liu, W., Zhong, H., Wang, R., Seeman, N.C.: Crystalline two-dimensional DNA-origami arrays. Angew. Chem. Int. Ed. **50**(1), 264–267 (2011). https://doi.org/10.1002/anie.201005911
4. Rothemund, P.W.K.: Folding DNA to create nanoscale shapes and patterns. Nature **440**(7082), 297–302 (2006). https://doi.org/10.1038/nature04586
5. Evans, C.G.: Crystals that count! Physical principles and experimental investigations of DNA tile self-assembly. PhD Thesis, California Institute of Technology (2014)
6. Gu, H., Chao, J., Xiao, S.-J., Seeman, N.C.: A proximity-based programmable DNA nanoscale assembly line. Nature **465**(7295), 202–205 (2010). https://doi.org/10.1038/nature09026

7. Lund, K., Manzo, A.J., Dabby, N., Michelotti, N., Johnson-Buck, A., Nangreave, J., Taylor, S., Pei, R., Stojanovic, M.N., Walter, N.G., Winfree, E., Yan, H.: Molecular robots guided by prescriptive landscapes. Nature **465**(7295), 206–210 (2010). https://doi.org/10.1038/nature09012
8. Lund, K., Manzo, A.T., Dabby, N., Micholotti, N., Johnson-Buck, A., Nangreave, J., Taylor, S., Pei, R., Stojanovic, M.N., Walter, N.G., Winfree, E., Yan, H.: Molecular robots guided by prescriptive landscapes. Nature **465**, 206–210 (2010)
9. Padilla, J.E., Sha, R., Kristiansen, M., Chen, J., Jonoska, N., Seeman, N.C.: A signal-passing DNA-strand-exchange mechanism for active self-assembly of DNA nanostructures. Angew. Chem. Int. Ed. **54**(20), 5939–5942 (2015)
10. Rothemund, P.W., Papadakis, N., Winfree, E.: Algorithmic self-assembly of DNA Sierpinski triangles. PLoS Biol. **2**(12), 424 (2004)
11. Woods, D., Doty, D., Myhrvold, C., Hui, J., Zhou, F., Yin, P., Winfree, E.: Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. Nature **567**(7748), 366–372 (2019)
12. Zhang, Y., McMullen, A., Pontani, L.-L., He, X., Sha, R., Seeman, N.C., Brujic, J., Chaikin, P.M.: Sequential self-assembly of DNA functionalized droplets. Nat. Commun. **8**(1), 21 (2017). https://doi.org/10.1038/s41467-017-00070-0
13. Fu, B., Patitz, M.J., Schweller, R.T., Sheline, R.: Self-assembly with geometric tiles. In: Czumaj, A., Mehlhorn, K., Pitts, A.M., Wattenhofer, R. (eds.) Automata, Languages, and Programming—39th International Colloquium, ICALP 2012, July 9–13, 2012, Proceedings, Part I. LNCS, vol. 7391, pp. 714–725. Springer, Warwick (2012)
14. Padilla, J.E., Patitz, M.J., Pena, R., Schweller, R.T., Seeman, N.C., Sheline, R., Summers, S.M., Zhong, X.: Asynchronous signal passing for tile self-assembly: Fuel efficient computation and efficient assembly of shapes. In: UCNC. Lecture Notes in Computer Science, vol. 7956, pp. 174–185. Springer, Milan (2013)
15. Winfree, E.: Algorithmic self-assembly of DNA. PhD Thesis, California Institute of Technology (1998)
16. Woods, D.: Intrinsic universality and the computational power of self-assembly. Philos. Trans. R. Soc. Lond. A Math. Phys. Eng. Sci. (2015). https://doi.org/10.1098/rsta.2014.0214
17. Soloveichik, D., Winfree, E.: Complexity of self-assembled shapes. SIAM J. Comput. **36**(6), 1544–1569 (2007)
18. Kao, M.-Y., Schweller, R.T.: Randomized self-assembly for approximate shapes. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP (1). Lecture Notes in Computer Science, vol. 5125, pp. 370–384. Springer, Reykjavik (2008)
19. Luchsinger, A., Schweller, R.T., Wylie, T.: Self-assembly of shapes at constant scale using repulsive forces. In: UCNC. Lecture Notes in Computer Science, vol. 10240, pp. 82–97. Springer, New York City (2017)
20. Patitz, M.J., Schweller, R.T., Summers, S.M.: Exact shapes and Turing universality at temperature 1 with a single negative glue. In: Cardelli, L., Shih, W.M. (eds.) DNA Computing and Molecular Programming—17th International Conference, DNA 17, September 19–23, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6937, pp. 175–189. Springer, Pasadena (2011)
21. Hader, D., Patitz, M.J.: Geometric tiles and powers and limitations of geometric hindrance in self-assembly. In: Proceedings of the 18th Annual Conference on Unconventional Computation and Natural Computation (UCNC 2019), Tokyo, Japan June 3–7, 2019, pp. 191–204 (2019)
22. Hendricks, J., Padilla, J.E., Patitz, M.J., Rogers, T.A.: Signal transmission across tile assemblies: 3D static tiles simulate active self-assembly by 2D signal-passing tiles. In: Soloveichik, D., Yurke, B. (eds.) DNA Computing and Molecular Programming. Lecture Notes in Computer Science, vol. 8141, pp. 90–104. Springer, Tempe (2013). https://doi.org/10.1007/978-3-319-01928-4_7
23. Doty, D., Lutz, J.H., Patitz, M.J., Schweller, R.T., Summers, S.M., Woods, D.: The tile assembly model is intrinsically universal. In: Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science. FOCS 2012, pp. 302–310 (2012)
24. Hendricks, J., Patitz, M.J., Rogers, T.A.: Universal simulation of directed systems in the abstract tile assembly model requires undirectedness. In: Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2016), New Brunswick, October 9–11, 2016, pp. 800–809 (2016)
25. Hader, D., Koch, A., Patitz, M.J., Sharp, M.: The impacts of dimensionality, diffusion, and directedness on intrinsic universality in the abstract tile assembly model. In: Chawla, S. (ed.) Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, January 5–8, 2020, pp. 2607–2624. SIAM, Salt Lake City (2020)

26. Hader, D., Patitz, M.J.: The impacts of dimensionality, diffusion, and directedness on intrinsic cross-model simulation in tile-based self-assembly. In: Etessami, K., Feige, U., Puppis, G. (eds.) 50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10–14, 2023, Paderborn. LIPIcs, vol. 261, pp. 71–17119. Schloss Dagstuhl—Leibniz-Zentrum für Informatik, Wadern, Germany (2023). Doi: https://doi.org/10.4230/LIPIcs.ICALP.2023.71
27. Rothemund, P.W.K., Winfree, E.: The program-size complexity of self-assembled squares (extended abstract). In: STOC'00: Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing, pp. 459–468. ACM, Portland (2000)
28. Meunier, P.-E., Patitz, M.J., Summers, S.M., Theyssier, G., Winslow, A., Woods, D.: Intrinsic universality in tile self-assembly requires cooperation. In: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA 2014), Portland, January 5–7, 2014, pp. 752–771 (2014)
29. Alumbaugh, J.C., Daymude, J.J., Demaine, E.D., Patitz, M.J., Richa, A.W.: Simulation of programmable matter systems using active tile-based self-assembly. In: International Conference on DNA Computing and Molecular Programming, pp. 140–158. Springer (2019)
30. Lathrop, J.I., Lutz, J.H., Patitz, M.J., Summers, S.M.: Computability and complexity in self-assembly. Theory Comput. Syst. **48**(3), 617–647 (2011)