2012 IEEE 53rd Annual Symposium on Foundations of Computer Science

The tile assembly model is intrinsically universal

David Doty*, Jack H. Lutz†, Matthew J. Patitz‡,
Robert T. Schweller§, Scott M. Summers¶, Damien Woods‖
*Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125, USA. ddoty@caltech.edu
†Computer Science, Iowa State University, Ames, IA 50011 USA. lutz@cs.iastate.edu
‡Computer Science and Computer Engineering, University of Arkansas, Fayetteville, AR 72701, USA. patitz@uark.edu
§Computer Science, University of Texas–Pan American, Edinburg, TX, 78539, USA. rtschweller@utpa.edu
¶Computer Science and Software Engineering, University of Wisconsin–Platteville, Platteville, WI 53818, USA. summerss@uwplatt.edu
‖Computer Science, California Institute of Technology, Pasadena, CA 91125, USA. woods@caltech.edu

***Abstract*—We prove that the abstract Tile Assembly Model (aTAM) of nanoscale self-assembly is *intrinsically* universal. This means that there is a single tile assembly system $\mathcal{U}$ that, with proper initialization, simulates any tile assembly system $\mathcal{T}$. The simulation is "intrinsic" in the sense that the self-assembly process carried out by $\mathcal{U}$ is exactly that carried out by $\mathcal{T}$, with each tile of $\mathcal{T}$ represented by an $m \times m$ "supertile" of $\mathcal{U}$. Our construction works for the full aTAM at any temperature, and it faithfully simulates the deterministic or nondeterministic behavior of each $\mathcal{T}$.**

Our construction succeeds by solving an analog of the cell differentiation problem in developmental biology: Each supertile of $\mathcal{U}$, starting with those in the seed assembly, carries the "genome" of the simulated system $\mathcal{T}$. At each location of a potential supertile in the self-assembly of $\mathcal{U}$, a decision is made whether and how to express this genome, i.e., whether to generate a supertile and, if so, which tile of $\mathcal{T}$ it will represent. This decision must be achieved using asynchronous communication under incomplete information, but it achieves the correct global outcome(s).

I. Introduction

Structural DNA nanotechnology, pioneered by Seeman in the 1980s [25], exploits the information-processing capabilities of nucleic acids to engineer complex structures and devices at the nanoscale. This is a very "hands-off" sort of engineering: The right molecules are placed in solution, and the structures and devices self-assemble spontaneously according to the principles of chemical kinetics. Controlling such self-assembly processes is an enormous technical challenge, but impressive progress has already been made. Regular arrays [29], polyhedra [13], fractal structures [11], [23], maps of the world [24], curved three-dimensional vases [12], DNA tweezers [30], logic circuits [20], neural networks [21], and molecular robots [17] are a few of the nanoscale objects that have self-assembled in successful laboratory experiments. Motivating future applications include smaller, faster, more energy-efficient computer chips, single-molecule detection, and in-cell diagnosis and treatment of disease.

Theoretical computer science became involved with structural DNA nanotechnology just before the turn of this century. In his 1998 Ph.D. thesis, Winfree introduced a mathematical model of DNA tile self-assembly and proved that this model is Turing-universal, i.e., that it can simulate any Turing machine [28]. This implies that nanoscale self-assembly can be algorithmically directed, and that extremely complex structures and devices can in principle be engineered by self-assembly. Rothemund and Winfree [22] subsequently refined this model slightly, formulating the *abstract Tile Assembly Model (aTAM)*. The (two-dimensional) aTAM is an idealized model of error-free self-assembly in two dimensions that has been extensively investigated.

Very briefly, a *tile* in the aTAM is a unit square with a kind and strength of "glue" on each of its sides. A

Doty was supported by a Computing Innovation Fellowship under NSF grant 1019343 and NSF grants CCF-1219274 and CCF-1162589.
Lutz was supported by NSF grants 0652569 and 1143830. Part of this work was done during a sabbatical at Caltech and the Isaac Newton Institute for Mathematical Sciences at the University of Cambridge.
Patitz was supported in part by NSF grant CCF-1117672.
Schweller was supported in part by NSF grant CCF-1117672.
Woods was supported by NSF grant 0832824, the Molecular Programming Project, and NSF grants CCF-1219274 and CCF-1162589.

0272-5428/12 $26.00 © 2012 IEEE
DOI 10.1109/FOCS.2012.76

tile assembly system $\mathcal{T}$ consists of a finite collection T of tile types (with infinitely many tiles of each type in T available), a *seed assembly* σ consisting of one or more tiles of types in T, and a *temperature* τ. Self-assembly proceeds from the seed assembly σ, with tiles of types in T successively and nondeterministically attaching themselves to the existing assembly. Two tiles placed next to each other *interact* if the glues on their abutting sides match, and a tile *binds* to an assembly if the total strength on all of its interacting sides is at least τ. A more complete description of the aTAM appears in Section II.

Our topic is the *intrinsic universality* of the abstract Tile Assembly Model. We now explain what this means, starting with what it does *not* mean. By Winfree's above-mentioned result, there is a tile assembly system $\mathcal{U}$ that simulates a universal Turing machine. This universal Turing machine, and hence $\mathcal{U}$, can simulate any tile assembly system $\mathcal{T}$ (in fact, there are various aTAM software simulators available, e.g., [19]). But this is only a *computational* simulation. It *tells* us what $\mathcal{T}$ does, but it does not actually *do* what $\mathcal{T}$ does. The task of a Turing machine is to perform a computation, and a universal Turing machine performs the same computation as a machine that it simulates. The task of a tile assembly system is to perform the process of self-assembly, so a universal tile assembly system should *perform* the same self-assembly process as a tile assembly system that it simulates. This is what is meant by intrinsic universality.

This paper proves that the abstract Tile Assembly Model is intrinsically universal.

This means that there is a single tile set U that, with proper initialization (calling the initialized system $\mathcal{U}$), simulates any tile assembly system $\mathcal{T}$. The simulation is "intrinsic" in the sense that the self-assembly process carried out by $\mathcal{U}$ is exactly that carried out by $\mathcal{T}$, with each tile of $\mathcal{T}$ represented by an $m \times m$ "supertile" of $\mathcal{U}$. Our construction works for the full aTAM at any temperature (the simulating system $\mathcal{U}$ uses temperature 2), and it faithfully simulates the deterministic or nondeterministic behavior of each $\mathcal{T}$. This notion was studied by Ollinger [18] and others [4], [5], [10], [14]–[16] in the context of cellular automata and Wang tiling,

Our construction succeeds by solving an analog of the cell differentiation problem in developmental biology: Each supertile of $\mathcal{U}$, starting with those in the seed assembly, carries a complete encoding of the simulated system $\mathcal{T}$ (the "genome" of $\mathcal{T}$) along each of its sides, which we call "supersides". (This genome accounts for most of the m tiles of $\mathcal{U}$ that appear on each superside. Additional tiles along the superside identify the glue of the simulated tile of $\mathcal{T}$ and support a variety of communication mechanisms.) At each location of a potential supertile a decision is made whether and how to express this genome, i.e., whether to generate a supertile and, if so, which tile of $\mathcal{T}$ it will represent. (This latter choice will be nondeterministic precisely insofar as $\mathcal{T}$ is nondeterministic at this location.) This decision depends on very limited local information, but it achieves the correct global outcome(s). The self-assembly of $\mathcal{U}$ is thus a "developmental process" in which "supertile differentiation" is governed by local communication, while the "genome" is passed intact from supertile to supertile.

Our construction uses three basic interacting primitives to carry out the asynchronous communication under imperfect information needed for supertile differentiation and genome copying. These mechanisms are called frames, crawlers, and probes. The *frame* of a potential supertile consists of four layers of tiles just inside each extant superside. This frame is used for communication with adjacent supersides, which may or may not exist. Much of its function is achieved by a symmetry-breaking "competition" at each corner. Our construction uses many types of *crawlers*, which are messengers that copy and carry various pieces of information from place to place in the supertile. The *probes* of a superside are used to communicate with the opposite superside, which may or may not exist. The challenge is to program all this activity without ever blocking a path that may later be needed for intra-supertile communication. This summary is greatly oversimplified. An overview of our construction is presented in Section IV, and the full construction is presented in [8].

Our result shows that the aTAM is universal for itself, without recourse to indirect simulations by Turing machines or other models that obscure important properties of the model. For example, our result shows that the tile assembly model is able to simulate local interactions between tiles, nondeterminism, and tile growth processes in general, all on a global scale. Thus our intrinsically universal tile set captures, in a well defined way, all properties of any tile assembly system.

Intrinsic universality, with its precise notion of "simulate", has applications to the theory of self-assembly. Firstly, a useful type of simulation is where one shows that for all aTAM systems $\mathcal{T}$ there exists a tile assembly system $\mathcal{T}'$ in some *other* self-assembly model that simulates $\mathcal{T}$. This style of $\forall\mathcal{T}, \exists\mathcal{T}'$-simulation has been used previously [1], [2], [6] and is useful when comparing the power of tile assembly models. However, combining such a simulation statement with the statement of our

main result gives the immediate corollary that there is a *single* set of tiles U in the other model that, when appropriately seeded, simulates *any* aTAM tile assembly system $\mathcal{T}$. Hence our main result automatically shows the existence of a single, very powerful, tile set in the other model, a seemingly strong statement. Secondly, and more speculatively, our result opens the possibility for new research directions in self-assembly. For example, taken together with the result in [9], we now know of two classes of tile assembly systems that exhibit intrinsic universality: the full aTAM (our main result), and the more restricted locally consistent systems [9]. This gives a kind of closure property for these classes of systems. In the field of cellular automata, the notion of intrinsic universality has led to the development of formal tools to classify models of computation in terms of their ability to simulate each other [5]. The intrinsically universal cellular automata sit at the top of this "quasi-order". As an example of a concrete application of this work, the notion of intrinsic universality has been used [3], [4] to show that various elementary cellular automata are strictly less powerful than others. Specifically, it was shown that the communication complexity of those systems is too low for them to exhibit intrinsic universality, and so there is a wide range of behaviors they can never achieve. Such statements crucially make use of the fact that intrinsic universality uses a tight notion of "simulate". In tile self-assembly, we currently have very few tools by which to compare the abilities of models; the main comparisons essentially boil down to comparing tile complexity, or establishing whether or not the system can simulate Turing machines and thus make arbitrary computable shapes. Both comparisons, especially the latter, are necessarily rather coarse for comparing the expressibility of models, and we hope that our result, and the notion of "simulate" that we use, can inspire the development of work that elucidates a fine-grained structure for self-assembly.

We conclude this introduction with a brief discussion of related work. The most recent precursor is [9], in which some of the present authors showed that a restricted submodel of the aTAM is intrinsically universal. This was an extensive, computationally expressive submodel of the aTAM, but its provisos (temperature 2, no glue mismatches, and no binding strengths exceeding the temperature) were artificially restrictive, awkward to justify on molecular grounds, and inescapable from the standpoint of that paper's proof technique. Our approach here is perforce completely different. Both papers code the simulated system's genome along the supersides, but the resemblance ends there. The frames, crawlers, and probes that we use here are new. (The "probe-like" structures in [9] are too primitive to work for simulating the full aTAM.)

As noted in [9], constructions of Soloveichik and Winfree [26] and Demaine, Demaine, Fekete, Ishaque, Rafalin, Schweller, and Souvaine [7] can be used to achieve versions of intrinsic universality for tile assembly at temperature 1, but this appears to be a severe restriction. Additionally, the latter paper uses a generalized version of the aTAM (i.e., "hierarchical" self-assembly or the "two-handed" aTAM) that has a mechanism for long-range communication that is lacking in the standard aTAM and that obviates the need for the distributed communication mechanisms we employ to build supertiles. Also discussed in [9] are studies of universality in Wang tiling [27] such as those by Lafitte and Weiss [14]–[16]. While these studies are very significant in the contexts of mathematical logic and computability theory, they are concerned with the *existence* of tilings with no mismatches, and not with any *process* of self-assembly. In particular, most attempts to adapt the constructions of Wang tiling studies (such as those in [14]–[16]) to self-assembly result in a tile assembly system in which many junk assemblies are formed due to incorrect nondeterministic choices being made that arrest any further growth and/or result in assemblies that are inconsistent with the desired output assembly. We therefore require novel techniques to ensure that the only produced assemblies are those that represent the intended result or valid partial progress toward it. Furthermore, techniques used in constructing intrinsically universal cellular automata do not carry over to the aTAM as the models have fundamental differences; in particular, when a tile is placed it remains in-place forever, whereas cellular automata cells can be reused indefinitely. In fact, many of the challenging issues in proving our result are related to the fact that tiles, once placed, can block each other and, of course, that self-assembly is a highly asynchronous and nondeterministic process.

II. Abstract Tile Assembly Model

This section gives a brief informal sketch of the abstract Tile Assembly Model (aTAM). See [8] for a formal definition of the aTAM.

A *tile type* is a unit square with four sides, each consisting of a *glue label* (often represented as a finite string) and a nonnegative integer *strength*. We assume a finite set T of tile types, but an infinite number of copies of each tile type, each copy referred to as a *tile*. An *assembly* (a.k.a., *supertile*) is a positioning of tiles on the integer lattice $\mathbb{Z}^2$; i.e., a partial function $\alpha : \mathbb{Z}^2 \dashrightarrow T$. Let $\mathcal{A}^T$ denote the set of all assemblies of tiles from

T, and let $\mathcal{A}^T_{<\infty}$ denote the set of finite assemblies of tiles from T. Write $\alpha \sqsubseteq \beta$ to denote that α is a *subassembly* of β, which means that $\mathrm{dom}\ \alpha \subseteq \mathrm{dom}\ \beta$ and $\alpha(p) = \beta(p)$ for all points $p \in \mathrm{dom}\ \alpha$. Two adjacent tiles in an assembly *interact* if the glue labels on their abutting sides are equal and have positive strength. Each assembly induces a *binding graph*, a grid graph whose vertices are tiles, with an edge between two tiles if they interact. The assembly is τ-*stable* if every cut of its binding graph has strength at least τ, where the weight of an edge is the strength of the glue it represents. That is, the assembly is stable if at least energy τ is required to separate the assembly into two parts.

A *tile assembly system* (TAS) is a triple $\mathcal{T} = (T, \sigma, \tau)$, where T is a finite set of tile types, $\sigma : \mathbb{Z}^2 \dashrightarrow T$ is a finite, τ-stable *seed assembly*, and τ is the *temperature*. An assembly α is *producible* if either $\alpha = \sigma$ or if β is a producible assembly and α can be obtained from β by the stable binding of a single tile. In this case write $\beta \rightarrow_1^{\mathcal{T}} \alpha$ (α is producible from β by the attachment of one tile), and write $\beta \rightarrow^{\mathcal{T}} \alpha$ if $\beta \rightarrow_1^{\mathcal{T}*} \alpha$ (α is producible from β by the attachment of zero or more tiles). When $\mathcal{T}$ is clear from context, we may write $\rightarrow_1$ and $\rightarrow$ instead. An assembly is *terminal* if no tile can be τ-stably attached to it. Let $\mathcal{A}[\mathcal{T}]$ be the set of producible assemblies of $\mathcal{T}$, and let $\mathcal{A}_\square[\mathcal{T}] \subseteq \mathcal{A}[\mathcal{T}]$ be the set of producible, terminal assemblies of $\mathcal{T}$. A TAS $\mathcal{T}$ is *directed* (a.k.a., *deterministic*, *confluent*) if $|\mathcal{A}_\square[\mathcal{T}]| = 1$.

We make the following assumptions that do not affect the fundamental capabilities of the model, but which will simplify our main construction. Since the behavior of a TAS $\mathcal{T} = (T, \sigma, \tau)$ is unchanged if every glue with strength greater than τ is changed to have strength exactly τ, we assume henceforth that all glue strengths are in the set $\{0, 1, \ldots, \tau\}$. We assume that glue labels are never shared between glues of unequal strength.

III. Main result

To state our main result, we must formally define what it means for one TAS to "simulate" another. We focus in particular on a sort of "direct simulation" via block replacement ($m \times m$ blocks of tiles in the simulating system represent single tiles in the simulated system). The intuitive goal of the following definition is identical to that in [9], and corrects some subtle errors there.

Let $m \in \mathbb{Z}^+$. An m-*block supertile* over tile set T is a partial function $\alpha : \mathbb{Z}_m \times \mathbb{Z}_m \dashrightarrow T$, where $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$. Let B^T_m be the set of all m-block supertiles over T. The m-block with no domain is said to be *empty*. For a general assembly $\alpha : \mathbb{Z}^2 \dashrightarrow T$ and $x, y \in \mathbb{Z}$, define $\alpha^m_{x,y}$ to be the m-block supertile defined by $\alpha^m_{x,y}(i, j) = \alpha(mx + i, my + j)$ for $0 \leq i, j < m$. A partial function $R : B^S_m \dashrightarrow T$ is said to be a *valid* m-*block supertile representation* from S to T if for any $\alpha, \beta \in B^S_m$ such that $\alpha \sqsubseteq \beta$ and $\alpha \in \mathrm{dom}\ R$, then $R(\alpha) = R(\beta)$.

For a given valid m-block supertile representation function R from tile set S to tile set T, define the *assembly representation function* $R^* : \mathcal{A}^S \rightarrow \mathcal{A}^T$ such that $R^*(\alpha') = \alpha$ if and only if $\alpha(x, y) = R(\alpha'^m_{x,y})$ for all $x, y \in \mathbb{Z}$.[1] 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 $\alpha'^m_{x,y}$, $(x + u, y + v) \in \mathrm{dom}\ \alpha$ for some $u, v \in \{-1, 0, 1\}$, or if α' has at most one non-empty m-block $\alpha^m_{0,0}$. In other words, α' may have tiles on supertile blocks representing empty space in α, but only if that position is adjacent to a tile in α.

A TAS $\mathcal{S} = (S, \sigma_S, \tau_S)$ *simulates* a TAS $\mathcal{T} = (T, \sigma_T, \tau_T)$ at scale $m \in \mathbb{Z}^+$ if there exists an m-block representation $R : B^S_m \rightarrow T$ such that the following hold:

1) Equivalent Production.
   a) $\{R^*(\alpha') | \alpha' \in \mathcal{A}[\mathcal{S}]\} = \mathcal{A}[\mathcal{T}]$.
   b) For all $\alpha' \in \mathcal{A}[\mathcal{S}]$, α' maps cleanly to $R^*(\alpha')$.
2) Equivalent Dynamics.
   a) If $\alpha \rightarrow^{\mathcal{T}} \beta$ for some $\alpha, \beta \in \mathcal{A}[\mathcal{T}]$, then for all α' such that $R^*(\alpha') = \alpha$, $\alpha' \rightarrow^{\mathcal{S}} \beta'$ for some $\beta' \in \mathcal{A}[\mathcal{S}]$ with $R^*(\beta') = \beta$.
   b) If $\alpha' \rightarrow^{\mathcal{S}} \beta'$ for some $\alpha', \beta' \in \mathcal{A}[\mathcal{S}]$, then $R^*(\alpha') \rightarrow^{\mathcal{T}} R^*(\beta')$.

Let REPR denote the set of all supertile representation functions (i.e., m-block supertile representation functions for some $m \in \mathbb{Z}^+$). Let $\mathfrak{C}$ be a class of tile assembly systems, and let U be a tile set.[2] Note that every element of $\mathfrak{C}$, REPR, and $\mathcal{A}^U_{<\infty}$ is a finite object, and hence can be represented in a suitable format for computation in some formal system such as Turing machines. We say U is *intrinsically universal* for $\mathfrak{C}$ if there are computable functions $\mathcal{R} : \mathfrak{C} \rightarrow \mathrm{REPR}$ and $S : \mathfrak{C} \rightarrow \mathcal{A}^U_{<\infty}$ and $\tau' \in \mathbb{Z}^+$ such that, for each $\mathcal{T} = (T, \sigma, \tau) \in \mathfrak{C}$, there is a constant $m \in \mathbb{N}$ such that, letting $R = \mathcal{R}(\mathcal{T})$, $\sigma_{\mathcal{T}} = S(\mathcal{T})$, and $\mathcal{U}_{\mathcal{T}} = (U, \sigma_{\mathcal{T}}, \tau')$, $\mathcal{U}_{\mathcal{T}}$ simulates $\mathcal{T}$ at scale m and using supertile representation function R. That is, $\mathcal{R}(\mathcal{T})$ outputs a representation function that interprets assemblies of $\mathcal{U}_{\mathcal{T}}$ as assemblies of $\mathcal{T}$, and $S(\mathcal{T})$ outputs the seed assembly used to program tiles from U to represent the seed assembly of $\mathcal{T}$.

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

[2] TAS's having tile set U are not necessarily elements of $\mathfrak{C}$, although this will be true in our main theorem since $\mathfrak{C}$ will be the set of all TAS's.

Our main theorem states that there is a single tile set capable of simulating any tile assembly system.

Theorem III.1. *There is a tile set U that is intrinsically universal for the class of all tile assembly systems.*

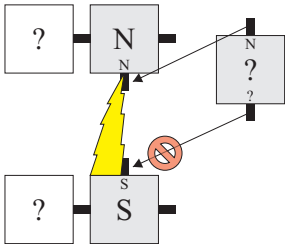
Our intrinsically universal tile set U works at temperature $\tau' = 2$. Throughout this paper, $\mathcal{T} = (T, \sigma, \tau)$ will denote an arbitrary TAS being simulated. Let $g \in \mathbb{Z}^+$ denote the number of different glues in T; note that $g = O(|T|)$. In our main construction, we achieve scale factor $m = O(g^4 \log g)$; an interesting open question is decreasing this scale factor or proving a nontrivial lower bound on it.

IV. High-level description of construction

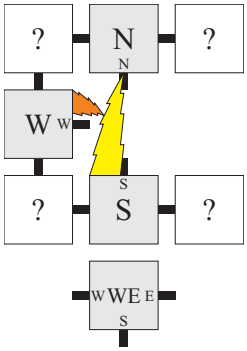
In the remainder of this extended abstract we sketch an intuitive overview of the construction. The full construction, including detailed figures, is contained in [8]. Let $\mathcal{T} = (T, \sigma, \tau)$ be a TAS being simulated by $\mathcal{U} = (U, \sigma_{\mathcal{T}}, 2)$, where U is the universal tile set and $\sigma_{\mathcal{T}}$ is the appropriate seed assembly for U to simulate $\mathcal{T}$. The seed assembly $\sigma_{\mathcal{T}}$ encodes information about the glues from $\mathcal{T}$ that are on the perimeter of σ, with each exposed tile-side of σ encoded as a "superside" . In particular, glues are simply encoded as binary strings of length $O(\log |T|)$. (Glue strengths are not explicitly encoded since their effect on binding is implicitly accounted for by other parts of the design.) Most importantly, each of these supersides, as well as each superside of all subsequently grown supertiles, encodes information about the entire TAS $\mathcal{T}$. This information is like the "genome" of the system that is transported to each supertile of the assembly in order to help direct its growth based on the contents of T.

A. The fundamental problem of simulating arbitrary tile systems

The basic problem faced by any superside adjacent to an empty supertile is this: the superside must determine what other superside(s) are adjacent to the same empty supertile, what glue(s) are on those sides, whether those glues are part of a tile type $t \in T$ and whether they have enough strength to bind t (and to choose among multiple tile types if more than one match the glues), and if so, the supersides on the remaining sides of t must be constructed (i.e., placed as "output" on empty supersides). This must be done in concert with other supersides that will be attempting the same thing, possibly "unaware" of each others' presence, and it must be done without prior knowledge of which other supersides will eventually arrive and the order and timing of their arrival.



(a) The north side has to talk with the south side, and the consensus is that there is no tile whose north side is 'N' and whose south side is 'S'. Better luck next time!



(b) Uh oh! Sometime later on, there seems to be no way to "communicate" from the west side to the east that the 'WE' tile should be represented here.

Fig. 1: How should supertiles communicate across a gap without "cutting the supertile in two"?

To illustrate the nontriviality of this problem, consider the following scenario illustrated in Figure 1a. Two supertiles arrive at positions that are north and south of an empty supertile position, with the east and west positions being unoccupied. The south superside has no choice but to attempt to "contact" the north superside, for it may be the case that their glues match that of some tile type $t \in T$, in which case the west and east supersides representing the sides of t must be put in place. But suppose that although there is a north superside, the glue it represents is not shared with the south glue on any tile type in T, or perhaps their combined strength is less than τ. (See Figure 1a.) Intuitively, it seems that to determine this, the north and south supersides must connect, in order to bring their glues together and do a computation/lookup to find that no tile type in T shares them. But once they have connected, the west and east sides of the supertile are now sealed off from each other.

Suppose that at a later time, a superside arrives on the west, and its encoded glue *is* shared with the west glue on some tile type $t \in T$ (with a north glue mismatching that of the supertile already present there; see Figure 1b). This means that t's east glue must now be represented by constructing an east output superside; however, this information cannot be communicated from the west side

of the supertile because the previous attempt to connect the south and north has created a barrier between east and west.

Note that such problems do not appear in Wang tiling constructions because the nondeterministic operator can simply guess what the other sides are.

This is not the only potential pitfall to be faced, but it illustrates an example of the difficulty of coordinating interaction between multiple supersides in the absence of knowledge about which supersides will eventually arrive, the order in which they will arrive, and what glues they will represent.[3]

B. Basic protocol

Here we give a high-level overview of our construction.

1) Frame: Each superside fills in a 4-layer "frame" in the supertile before doing anything else. The purpose of the frame is to give each superside as much information as possible about the other available supersides, to help coordinate their interaction. Each superside attempts to "become an input superside" of the supertile by competing to place a single tile at a particular position on its left end, and another on its right end. It is competing with a (potential) adjacent superside near their common corner; for example, the south superside competes on its left end with the west superside (at the southwest corner) and on its right end with the east superside (at the southeast corner). Therefore there are four competitions, one at each corner, and for each corner there is both a winner superside and a loser superside. The "loser" may simply be a superside that is not present and never will be, or it may be a superside that is present but lost the competition because it arrived later. Corner tiles initiate the growth of the first (outermost) layer of the frame, and it is by the initiation from a corner that it lost that a losing side gains the information that there was an adjacent side to compete with. For corners that it won, it cannot know whether an adjacent superside is present and lost, or whether there is simply no adajacent superside.[4]

The first layer of the frame grows from the corners of the superside to its center, at which point the entire superside knows whether it won or lost each corner. The subsequent layers of the frame allow the pattern of wins and losses for each side to be propagated among adjacent sides in a well defined way. The careful design of the frame and the algorithm for passing this information allows the large set of possible scenarios (of all subsets of sides which may be present in all possible orderings) to be condensed into a much smaller set of equivalent classes of scenarios which can then be properly handled by the next portions of the supertile to grow (as necessary) from the frame: "crawlers" which grow along adjacent supersides, and "probes" which grow across the centers of supertile spaces attempting to communicate with opposite supersides (if they exist). It is notable that, as the frame grows, it propagates all information from the superside of the adjacent supertile (which is acting as an output side for that supertile that initiates the formation of this potential input side for a new supertile), including the encoding of the glue on that superside and the encoding of the system $\mathcal{T}$ (i.e. the "genome"), into the interior of the new supertile.

Reference [8] details the algorithm used to grow the frame to achieve this gathering of information.

2) Crawlers and lookup tables: Once the frame has formed, information about the glues represented by the supersides, the simulated tile system $\mathcal{T}$, and the win/loss status of each side (possibly along with information about additional sides) is presented on the inside of the frame. At this point, pairs of adjacent edges (i.e. those sharing a corner) may initiate the growth of a "crawler" component. (The determination of whether or not a particular pair will do so is discussed later.) At a high level, when a crawler is initiated it contains the encoding of the glue for one superside, and as it forms it grows across the adjacent superside, gathering the information of the glue on that side as it grows across it. Next, it grows across the encoding of a "tile lookup table" which is an encoding of the tile set T that allows it to determine if the glues that it has collected so far match a tile in $t \in T$ and have sufficient strength for t to bind. If so, then this supertile should simulate t and the crawler's job becomes to grow around the remaining sides of the supertile and to create output supersides for those sides which aren't already occupied by input supersides.

Of course, this is an ideal scenario; what could go wrong? Perhaps the glues do not match a tile type. Perhaps another superside arrived while we were attempting to output on that side. Perhaps there are only supersides on the south and north, and they must reach across the gap of the empty supertile between them to cooperate and place east and west output supersides.

[3]These problems would be easier (if cumbersome) to overcome by growing in three dimensions, but achieving a planar construction is nontrivial. Furthermore, since two is the standard number of dimensions in tile assembly, a planar construction is required if we want our result to be most applicable to other (future) results in the abstract tile assembly model, as well as in the wide spectrum of other tile assembly models.

[4]Parallel programmers may be reminded of a similar phenomenon: a thread locking a mutex does not know whether other threads will eventually attempt to access it, but a thread encountering a locked mutex knows for sure that another thread is currently accessing it.

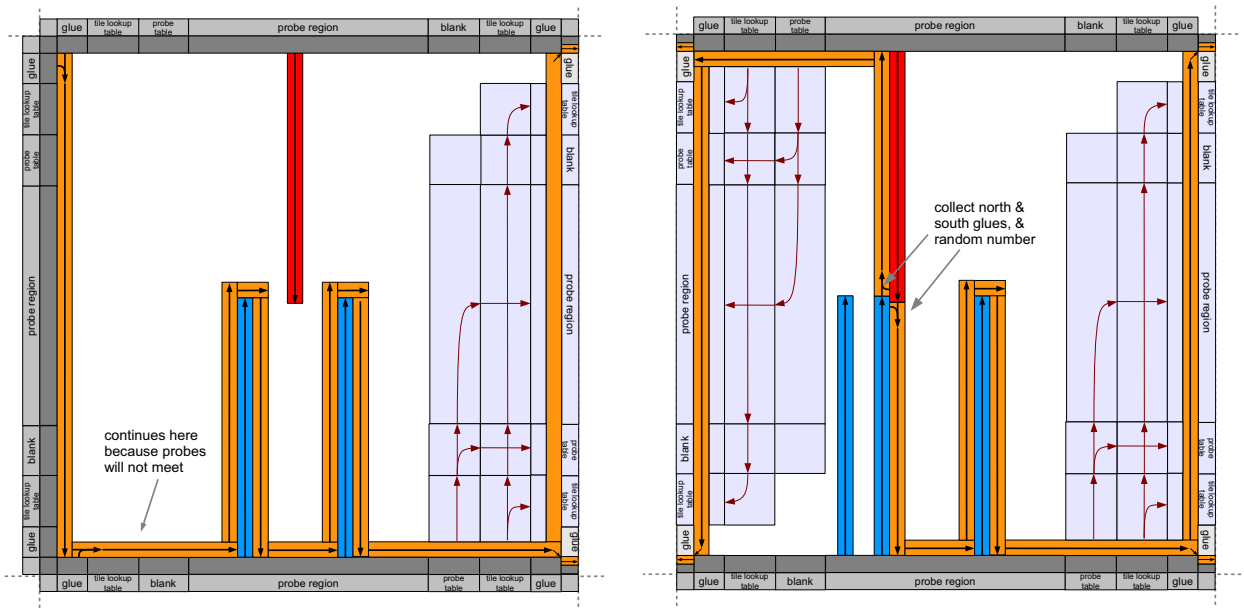
More generally, a crawler crawls around a supertile "collecting" input glues. It starts in an `unfilled` state until a tile lookup reveals that it has collected glues with sufficient strength to place an output tile type; at this point the crawler enters a `full` state. However, it has not yet committed to creating an output tile type because there may be other crawlers present that are also `full`. Some symmetry-breaking is used to determine when a `full` crawler changes to an `output` state and takes responsibility for determining the output tile type and placing output supersides. Our main goal in justifying the correctness of the construction is proving that if subsets of supersides represent glues that are sufficient to bind a tile, then eventually exactly one crawler will enter the `output` state and decide the output tile type t (or two crawlers in the special case where they originate from meeting probes and are guaranteed to make the same output decision).

3) More general crawler protocol: The more general protocol followed by crawlers is this. Whenever two supersides "connect", at a corner as in Figure 2a, or by reaching across the gap as in Figure 2b, they (sometimes, depending on information supplied by the frame) initiate a crawler that first combines their glues and does a lookup to see if a tile matches these glues. Crawlers always move counterclockwise around a supertile. The general rule is: *Initiate a crawler when two sides meet, unless we have enough information from the frame to see that another crawler will be on its way from another corner.* Note that sometimes two crawlers are initiated because the "later" crawler (the crawler in the more counterclockwise direction) does not "know" (based on only its two adjacent sides) about the first crawler. If the lookup is successful, the crawler becomes `full` and will attempt to place output supersides if there are potentially empty supersides. On each potential output superside, the crawler first "tests" to see if an output side is already present, only outputting if necessary. If the lookup is unsuccessful (i.e., the glues available to the crawler were not sufficient to bind a tile), the `unfilled` crawler crawls to the edge of the supertile to wait for a potential new input superside to arrive. If this superside ever does arrive, it will initiate its own crawler that will combine the information from the first crawler (and its two glues), to see if all *three* glues are sufficient by performing a new lookup. This new crawler will follow the same protocol.

4) Multiple crawlers: As previously mentioned, there are some situations in which two crawlers may be initiated and begin growth. In such a situation, a crawler c_1 may arrive at a side to find that another crawler c_2 has already begun growth from there; if so, c_1 crawls over the "back" of c_2 to see if c_2 became `full` (i.e., had a successful table lookup). If c_2 is `full`, c_1 stops, allowing c_2 to take responsibility for outputting. Otherwise, c_1 does its own lookup using all glues (whatever glues that c_1 has already collected before encountering c_2, plus the new glue on the side that initiated the growth of c_2). It is possible for c_1 to receive this additional information from c_2 because crawlers pass all collected and computed information up through themselves. This is necessary for supporting such "piggybacking" crawlers, as well as making the necessary information available when it becomes time to create output supersides. In this case c_1 may overtake c_2 to place output.

In cases where all four supersides are present, although the `output` crawler does not need to deposit output supersides, it must still decide on an output tile type so that the representation function can uniquely decode which tile type is represented by the supertile. In this case, it may be the case that two crawlers exist but one of them does not run into the other. However, we still require symmetry-breaking so that only one of them changes to the `output` state. In this case, once a crawler has encountered the fourth superside (which happens after it has traversed the full length of two supersides, it has complete information (gathered from the frame) about the win-loss configuration and therefore knows whether another crawler was independently initiated. In this case, a precedence ranking on corners that initiate crawlers (NW $>$ SW $>$ SE $>$ NE) is used to determine whether to transition to the `output` state or to simply die (in effect, letting the other, higher-precedence crawler become the unique `output` crawler).

5) Probes: "Probes" are used for communication across a gap between two potential input supersides on opposite sides (i.e. north and south or east and west) when necessary. Suppose the south superside needs to communicate with the north superside. Recall that the south superside's frame either won or lost on each end of the superside. If either end lost, then this means there is a supertile adjacent to both south and north (west if south lost in the southwest corner, and east if south lost in the southeast corner). Therefore there is no need for probes, since crawlers will eventually connect the south glue with the north glue. Only if the south superside is "win-win" does it send probes to potentially connect with the north superside. Since all supersides follow this rule, this ensures that at most two sides ever grow probes, and if so, then they are opposite sides (since adjacent sides cannot both be win-win). This ensures that orthogonal



(a) Three-sided binding with input supersides on the north, west and south (the scenario described in Fig. 1). A single (orange) crawler is initiated from the north-west corner and outputs to the east. North and south grow probes, that do not meet.

(b) Two-sided binding for two opposite sides on north and south. North and south grow probes (red and blue), which meet and initiate two (orange) crawlers. These crawlers output to the east and west respectively.

Fig. 2: Two examples of how probes and crawlers work together to enable cooperative binding of supersides.

probes cannot grow and interfere with each other.

The design of probes ensures that they "close the gap" (connect two sides of the supertile) if and only if their supersides represent glues that map to a tile type $t \in T$ and have sufficient strength for t to bind. The probes grow from a region on the superside known as the "probe region". Each glue in T has its own unique subregion in the probe region . The supersides do not grow probes symmetrically: north and east grow probes in one way, and west and south grow them in a complementary way. Suppose the glue on the north is n and the glue on the south is s. The north superside will grow a probe in the subregion associated with n. For every glue g that has the property that there is some tile type $t \in T$ with g on the north, s on the south, and g and s have combined strength at least τ, the south superside will grow a probe in the subregion associated with that g. If no tile type matches glue s on the south and n on the north (or if n and s have insufficient combined strength), but both north and south probes form, they will be guaranteed to leave sufficiently wide gaps for crawlers, which may be initiated and arrive later, to make their way around and between the probes . This is because each probe subregion is at least $\Omega(|T|)$ tiles from its adjacent probe subregions, but crawlers are only $O(\log |T|)$ tiles wide. If the probes do meet in the middle of the supertile (indicating that a tile type $t \in T$ matches the north/south glues and has sufficient strength to bind), they initiate their own crawlers which can place the output supersides representing the west and east sides of t.

6) Simulation of nondeterministic tile systems: In each of these cases, if the simulated system $\mathcal{T}$ is nondeterministic, there may be more than one tile type that matches a given set of input glues. To handle this scenario, a "random number" is produced through nondeterministic attachment of tile types to a special "random number selector component" and used as an index to select one of the possible tile types. (A similar mechanism was used in [9].) It is crucial that if two probes cut off two sides of a supertile from each other, each side's crawlers must use the same random number to select the tile type to output, or else they may choose differently and place output glues that are not consistent with any single tile type in T. This is why probes generate a random number and advertise it to each side of the probe. However, if probes do not meet, then eventually a single crawler will be responsible for choosing an output tile type, so it is sufficient for the crawler to generate a random number just before it begins a tile lookup.

The full details of the construction are described in reference [8].

ACKNOWLEDGEMENT

We would like to thank Matthew Cook for pointing out a simplification to our construction, and David Soloveichik for stimulating conversations.

REFERENCES

[1] Leonard M. Adleman, Jarkko Kari, Lila Kari, Dustin Reishus, and Petr Sosík, *The undecidability of the infinite ribbon problem: Implications for computing by self-assembly*, SIAM Journal on Computing **38** (2009), no. 6, 2356–2381, Preliminary version appeared in FOCS 2002.

[2] Sarah Cannon, Erik D. Demaine, Martin L. Demaine, Sarah Eisenstat, Matthew J. Patitz, Robert Schweller, Scott M. Summers, and Andrew Winslow, *Two hands are better than one (up to constant factors)*, Arxiv preprint arXiv:1201.1650 (2012).

[3] Eric Goles Ch., Cedric Little, and Ivan Rapaport, *Understanding a non-trivial cellular automaton by finding its simplest underlying communication protocol*, ISAAC 2008, The 19th International Symposium on Algorithms and Complexity, Lecture Notes in Computer Science, vol. 5369, 2008, pp. 592–604.

[4] Eric Goles Ch., Pierre-Etienne Meunier, Ivan Rapaport, and Guillaume Theyssier, *Communication complexity and intrinsic universality in cellular automata*, Theor. Comput. Sci. **412** (2011), no. 1-2, 2–21.

[5] Marianne Delorme, Jacques Mazoyer, Nicolas Ollinger, and Guillaume Theyssier, *Bulking II: Classifications of cellular automata*, Theor. Comput. Sci. **412** (2011), no. 30, 3881–3905.

[6] Erik D. Demaine, Martin L. Demaine, Sandor Fekete, Matthew J. Patitz, Robert Schweller, Andrew Winslow, and Damien Woods, *One tile to simulate any tile system*, Preprint (2012).

[7] Erik D. Demaine, Martin L. Demaine, Sándor P. Fekete, Mashhood Ishaque, Eynat Rafalin, Robert T. Schweller, and Diane L. Souvaine, *Staged self-assembly: Nanomanufacture of arbitrary shapes with $O(1)$ glues*, Natural Computing **7** (2008), no. 3, 347–370, Preliminary version appeared in DNA 13.

[8] David Doty, Jack H. Lutz, Matthew J. Patitz, Robert T. Schweller, Scott M. Summers, and Damien Woods, *The tile assembly model is intrinsically universal*, Tech. Report 1111.3097, Computing Research Repository, 2012.

[9] David Doty, Jack H. Lutz, Matthew J. Patitz, Scott M. Summers, and Damien Woods, *Intrinsic universality in self-assembly*, STACS 2010: Proceedings of The Twenty-Seventh International Symposium on Theoretical Aspects of Computer Science, Leibniz International Proceedings in Informatics (LIPIcs), vol. 5, 2010, pp. 275–286.

[10] B. Durand and Zs. Róka, *The game of life: universality revisited*, Cellular Automata (M. Delorme and J. Mazoyer, eds.), Kluwer, 1999.

[11] Kenichi Fujibayashi, Rizal Hariadi, Sung Ha Park, Erik Winfree, and Satoshi Murata, *Toward reliable algorithmic self-assembly of DNA tiles: A fixed-width cellular automaton pattern*, Nano Letters **8** (2007), no. 7, 1791–1797.

[12] D. Han, S. Pal, J. Nangreave, Z. Deng, Y. Liu, and H. Yan, *DNA origami with complex curvatures in three-dimensional space*, Science **332** (2011), no. 6027, 342.

[13] Y. He, T. Ye, M. Su, C. Zhang, A.E. Ribbe, W. Jiang, and C. Mao, *Hierarchical self-assembly of DNA into symmetric supramolecular polyhedra*, Nature **452** (2008), no. 7184, 198–201.

[14] Grégory Lafitte and Michael Weiss, *Universal tilings*, STACS 2007, 24th Annual Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, February 22-24, 2007, Proceedings (Wolfgang Thomas and Pascal Weil, eds.), Lecture Notes in Computer Science, vol. 4393, Springer, 2007, pp. 367–380.

[15] ———, *Simulations between tilings*, Tech. report, University of Athens, 2008.

[16] ———, *An almost totally universal tile set*, Theory and Applications of Models of Computation, 6th Annual Conference, TAMC 2009, Changsha, China, May 18-22, 2009. Proceedings (Jianer Chen and S. Barry Cooper, eds.), Lecture Notes in Computer Science, vol. 5532, Springer, 2009, pp. 271–280.

[17] Kyle Lund, Anthony T. Manzo, Nadine Dabby, Nicole Micholotti, Alexander Johnson-Buck, Jeanetter Nangreave, Steven Taylor, Renjun Pei, Milan N. Stojanovic, Nils G. Walter, Erik Winfree, and Hao Yan, *Molecular robots guided by prescriptive landscapes*, Nature **465** (2010), 206–210.

[18] Nicolas Ollinger, *Intrinsically universal cellular automata*, CSP, 2008, pp. 199–204.

[19] Matthew J. Patitz, *Simulation of self-assembly in the abstract tile assembly model with ISU TAS*, FNANO 2009: 6th Annual Conference on Foundations of Nanoscience: Self-Assembled Architectures and Devices (Snowbird, Utah, USA, April 20-24 2009), Sciencetechnica, 2009, pp. 209–219.

[20] L. Qian and E. Winfree, *Scaling up digital circuit computation with DNA strand displacement cascades*, Science **332** (2011), no. 6034, 1196.

[21] L. Qian, E. Winfree, and J. Bruck, *Neural network computation with dna strand displacement cascades*, Nature **475** (2011), no. 7356, 368–372.

[22] Paul W. K. Rothemund and Erik Winfree, *The program-size complexity of self-assembled squares (extended abstract)*, STOC 2000: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, 2000, pp. 459–468.

[23] Paul W.K. Rothemund, Nick Papadakis, and Erik Winfree, *Algorithmic self-assembly of DNA Sierpinski triangles*, PLoS Biology **2** (2004), no. 12, 2041–2053.

[24] PW Rothemund, *Folding DNA to create nanoscale shapes and patterns*, Nature **440** (2006), no. 7082, 297–302.

[25] Nadrian C. Seeman, *Nucleic-acid junctions and lattices*, Journal of Theoretical Biology **99** (1982), 237–247.

[26] David Soloveichik and Erik Winfree, *Complexity of self-assembled shapes*, SIAM Journal on Computing **36** (2007), no. 6, 1544–1569, Preliminary version appeared in DNA 10.

[27] Hao Wang, *Proving theorems by pattern recognition – II*, The Bell System Technical Journal **XL** (1961), no. 1, 1–41.

[28] Erik Winfree, *Algorithmic self-assembly of DNA*, Ph.D. thesis, California Institute of Technology, June 1998.

[29] Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman, *Design and self-assembly of two-dimensional DNA crystals.*, Nature **394** (1998), no. 6693, 539–44.

[30] B. Yurke, A.J. Turberfield, A.P. Mills Jr, F.C. Simmel, and J.L. Neumann, *A DNA-fuelled molecular machine made of DNA*, Nature **406** (2000), no. 6796, 605–608.