An Introduction to Tile-Based Self-Assembly

Matthew J. Patitz*

University of Arkansas

Abstract. In this tutorial, we give a brief introduction to the field of tile-based algorithmic self-assembly. We begin with a description of Winfree's abstract Tile Assembly Model (aTAM) and a few basic exercises in designing tile assembly systems. We then survey a series of results in the aTAM. Next, we introduce the more experimentally realistic kinetic Tile Assembly Model (kTAM) and provide an exercise in error correction within the kTAM, then an overview of kTAM results . We next introduce the 2-Handed Assembly Model (2HAM), which allows entire assemblies to combine with each other in pairs, along with an exercise in developing a 2HAM system, and then give overviews of a series of 2HAM results. Finally, we briefly introduce a wide array of more recently developed models and discuss their various tradeoffs in comparison to the aTAM and each other.

1 Introduction

Self-assembly is the process by which a collection of relatively simple components, beginning in a disorganized state, spontaneously and without external guidance coalesce to form more complex structures. The process is guided by only local interactions between the components, which typically follow a basic set of rules. Despite the seemingly simplistic nature of self-assembly, its power can be harnessed to form structures of incredible complexity and intricacy. In fact, self-assembling systems abound in nature, resulting in everything from the delicate crystalline structure of snowflakes to many of the structurally and functionally varied components of biological systems.

Beyond the purely mathematically interesting properties of self-assembling systems, such systems have been recognized as an excellent template for the fabrication of artificial structures on the nanoscale. In order to precisely manipulate matter on the scale of individual atoms and molecules, several artificial self-assembling systems have been designed. Among these is the Tile Assembly Model introduced by Erik Winfree in his 1998 PdD thesis [44]. Formulated in two basic versions, the abstract Tile Assembly Model (aTAM) and the kinetic Tile Assembly Model (kTAM), it was based on a cross between the theoretical study of Wang tiles [43] (flat squares with labels on their edges) and novel DNA complexes being synthesized within Ned Seeman's laboratory [39]. The aTAM

* Department of Computer Science and Computer Engineering, University of Arkansas, Fayetteville, AR, USA mpatitz@self-assembly.net This author's research was supported in part by National Science Foundation Grant CCF-1117672.

provides a more high-level abstraction which ignores the possibility of errors and provides a framework for theoretical studies of the mathematical boundaries to the powers of such systems. The kTAM, on the other hand, injects more of the physical reality of the chemical kinetics into the model and allows for the study of the causes of errors and potential mechanisms for detecting, preventing, and/or correcting them.
We will first introduce the aTAM, giving relevant definitions and example aTAM systems. Next we will present a series a basic exercises in designing systems, intended to foster an interactive environment that provides the audience with a more firm understanding of the model. After this we will present a quick survey of results based on the aTAM, roughly sketching out what work has already been done and pointing out several open avenues for future research.
In the second main portion of the tutorial, we will introduce the kTAM and provide an explanation of relevant definitions and formulas. We will then conduct an interactive exercise in designing a kTAM system for basic error prevention. Next we will survey a series of result based on the kTAM to provide of picture of the progress that has been made. We will then introduce the 2-Handed Assembly Model (2HAM), in which, rather than requiring seeded assemblies which can grow only one tile at a time, arbitrarily large assemblies are allowed to combine with each other two at a time. We will conduct an exercise in designing a 2HAM system and then discuss a variety of 2HAM results, emphasizing especially those which provide comparisons and contrasts with the aTAM.
The third main portion of the tutorial will be comprised of very high-level introductions to a wide array of newer, derivative models. Such models have been introduced for a variety of reasons: to provide greater resilience to errors, to potentially provide more feasible laboratory implementations, to overcome theoretical limitations of the base models, to more faithfully mimic the behavior of given natural (especially biological) self-assembling systems, or simply to more fully explore the vast landscape of alternatives. Examples of such models include: temperature and concentration programming, the Staged Assembly Model, the Geometric Tile Assembly Model, and the Signal passing Tile Assembly Model.
The goal is to provide participants with a solid understanding of the original Tile Assembly Model, and then a brief overview of several newer models of tile-based self-assembly, as well as a high-level survey of the current state of results and open questions. Special care will be taken to try to show connections between results across models as well as potential relationships with other areas of research, in the hope of providing a basis for future projects linking ideas and results from self-assembly to those of various other areas of theoretical research.

2 Preliminaries and notation

In this section we provide a set of definitions and conventions that are used throughout this paper.
We work in the 2-dimensional discrete space $\mathbb{Z}^2$. Define the set $U_2 = \{(0,1),(1,0),(0,-1),(-1,0)\}$ to be the set of all *unit vectors* in $\mathbb{Z}^2$. We

also sometimes refer to these vectors by their cardinal directions N, E, S, W, respectively. All *graphs* in this paper are undirected. A *grid graph* is a graph $G = (V, E)$ in which $V \subseteq \mathbb{Z}^2$ and every edge $\{\boldsymbol{a}, \boldsymbol{b}\} \in E$ has the property that $\boldsymbol{a} - \boldsymbol{b} \in U_2$.

Intuitively, a tile type t is a unit square that can be translated, but not rotated, having a well-defined "side $\boldsymbol{u}$" for each $\boldsymbol{u} \in U_2$. Each side $\boldsymbol{u}$ of t has a "glue" with "label" $\text{label}_t(\boldsymbol{u})$–a string over some fixed alphabet– and "strength" $\text{str}_t(\boldsymbol{u})$–a nonnegative integer–specified by its type t. Two tiles t and t' that are placed at the points $\boldsymbol{a}$ and $\boldsymbol{a} + \boldsymbol{u}$ respectively, *bind* with *strength* $\text{str}_t(\boldsymbol{u})$ if and only if $(\text{label}_t(\boldsymbol{u}), \text{str}_t(\boldsymbol{u})) = (\text{label}_{t'}(-\boldsymbol{u}), \text{str}_{t'}(-\boldsymbol{u}))$.

In the subsequent definitions, given two partial functions f, g, we write $f(x) = g(x)$ if f and g are both defined and equal on x, or if f and g are both undefined on x.

Fix a finite set T of tile types. A *T-assembly*, sometimes denoted simply as an *assembly* when T is clear from the context, is a partial function $\alpha : \mathbb{Z}^2 \dashrightarrow T$ defined on at least one input, with points $\boldsymbol{x} \in \mathbb{Z}^2$ at which $\alpha(\boldsymbol{x})$ is undefined interpreted to be empty space, so that $\text{dom } \alpha$ is the set of points with tiles. We write $|\alpha|$ to denote $|\text{dom } \alpha|$, and we say α is *finite* if $|\alpha|$ is finite. For assemblies α and α', we say that α is a *subassembly* of α', and write $\alpha \sqsubseteq \alpha'$, if $\text{dom } \alpha \subseteq \text{dom } \alpha'$ and $\alpha(\boldsymbol{x}) = \alpha'(\boldsymbol{x})$ for all $x \in \text{dom } \alpha$.

2.1 Simulation Software

Throughout this tutorial, the Iowa State University Tile Assembly Simulator (ISU TAS) [32] will be used to present examples as well as to work on the exercises. The simulator and its source code are available online at `http://www.self-assembly.net` and it can be compiled for Windows, Mac OS, and linux. A brief tutorial on the use of the simulator will be provided during the tutorial, and more documentation can be downloaded along with the software.

3 The abstract Tile Assembly Model (aTAM)

3.1 Model definition

We now give a brief intuitive sketch of the abstract TAM. See [26, 37, 38, 44] for other developments of the model.

Given a set T of tile types, an *assembly* is a partial function $\alpha : \mathbb{Z}^2 \dashrightarrow T$. An assembly is *$\tau$-stable* if it cannot be broken up into smaller assemblies without breaking bonds of total strength at least τ, for some $\tau \in \mathbb{N}$.

Self-assembly begins with a *seed assembly* σ and proceeds asynchronously and nondeterministically, with tiles adsorbing one at a time to the existing assembly in any manner that preserves τ-stability at all times. A *tile assembly system* (*TAS*) is an ordered triple $\mathcal{T} = (T, \sigma, \tau)$, where T is a finite set of tile types, σ is a seed assembly with finite domain, and $\tau \in \mathbb{N}$. In this paper we deal exclusively with tile assembly systems in

which $\tau = 2$. A *generalized tile assembly system* (*GTAS*) is defined similarly, but without the finiteness requirements. We write $\mathcal{A}[\mathcal{T}]$ for the set of all assemblies that can arise (in finitely many steps or in the limit) from $\mathcal{T}$. An assembly $\alpha \in \mathcal{A}[\mathcal{T}]$ is *terminal*, and we write $\alpha \in \mathcal{A}_\square[\mathcal{T}]$, if no tile can be τ-stably added to it. It is clear that $\mathcal{A}[\mathcal{T}] \subseteq \mathcal{A}_\square[\mathcal{T}]$.
An assembly sequence in a TAS $\mathcal{T}$ is a (finite or infinite) sequence $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots)$ of assemblies in which each α_{i+1} is obtained from α_i by the addition of a single tile. The *result* $\text{res}(\boldsymbol{\alpha})$ of such an assembly sequence is its unique limiting assembly. (This is the last assembly in the sequence if the sequence is finite.) The set $\mathcal{A}[\mathcal{T}]$ is partially ordered by the relation $\longrightarrow$ defined by

$$\alpha \longrightarrow \alpha' \text{ iff there is an assembly sequence } \boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots) \text{ such that } \alpha_0 = \alpha \text{ and } \alpha' = \text{res}(\boldsymbol{\alpha}).$$

We say that $\mathcal{T}$ is *directed* (a.k.a. *deterministic*, *confluent*, *produces a unique assembly*) if the relation $\longrightarrow$ is directed, i.e., if for all $\alpha, \alpha' \in \mathcal{A}[\mathcal{T}]$, there exists $\alpha'' \in \mathcal{A}[\mathcal{T}]$ such that $\alpha \longrightarrow \alpha''$ and $\alpha' \longrightarrow \alpha''$. It is easy to show that $\mathcal{T}$ is directed if and only if there is a unique terminal assembly $\alpha \in \mathcal{A}[\mathcal{T}]$ such that $\sigma \longrightarrow \alpha$.
In general, even a directed TAS may have a very large (perhaps uncountably infinite) number of different assembly sequences leading to its terminal assembly. This seems to make it very difficult to prove that a TAS is directed. Fortunately, Soloveichik and Winfree [41] have recently defined a property, *local determinism*, of assembly sequences and proven the remarkable fact that, if a TAS $\mathcal{T}$ has *any* assembly sequence that is locally deterministic, then $\mathcal{T}$ is directed. Intuitively, an assembly sequence $\boldsymbol{\alpha}$ is locally deterministic if (1) each tile added in $\boldsymbol{\alpha}$ "just barely" binds to the existing assembly; (2) if a tile of type t_0 at a location $\boldsymbol{m}$ and its immediate "output-neighbors" are deleted from the *result* of $\boldsymbol{\alpha}$, then no tile of type $t \neq t_0$ can attach itself to the thus-obtained configuration at location $\boldsymbol{m}$; and (3) the result of $\boldsymbol{\alpha}$ is terminal.
A set $X \subseteq \mathbb{Z}^2$ *weakly self-assembles* if there exists a TAS $\mathcal{T} = (T, \sigma, \tau)$ and a set $B \subseteq T$ such that $\alpha^{-1}(B) = X$ holds for every terminal assembly $\alpha \in \mathcal{A}_\square[\mathcal{T}]$. Essentially, weak self-assembly can be thought of as the creation of a pattern of tiles from B (usually taken to be a unique "color") on a possibly larger "canvas" of un-colored tiles.
A set X *strictly self-assembles* if there is a TAS $\mathcal{T}$ for which every assembly $\alpha \in \mathcal{A}_\square[\mathcal{T}]$ satisfies $\text{dom } \alpha = X$. Essentially, strict self-assembly means that tiles are only placed in positions defined by the shape. Note that if X strictly self-assembles, then X weakly self-assembles. (Let all tiles be in B.)
Tiles are often depicted as squares whose various sides contain 0, 1, or 2 attached black squares, indicating whether the glue strengths on these sides are 0, 1, or 2, respectively. Thus, for example, a tile of the type shown in Figure 1 has glue of strength 0 on the left and bottom, glue of color 'a' and strength 2 on the top, and glue of color 'b' and strength 1 on the right. This tile also has a label 'L', which plays no formal role but may aid our understanding and discussion of the construction.

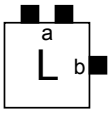
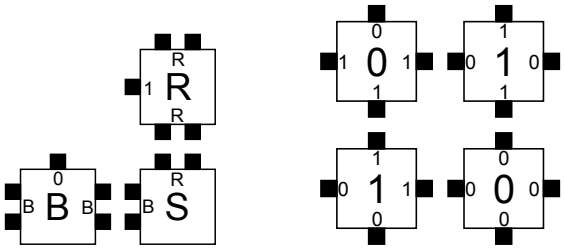


Fig. 1: An example tile type.

3.2 Examples and exercises

Here we present a basic example of an aTAM system, followed by a few suggested exercises in designing aTAM systems.

Example: a binary counter The aTAM is capable of Turing universal computation, so our first example will consist of a system which self-assembles a simple computation, namely an infinite binary counter. Figure 2a shows three tile types which will be used to form the boundary of the counter on its bottom and right sides. Figure 2b shows the additional 4 tile types needed to perform the actual counting and to display, via their labels, the current binary number. We will define our binary counter tile assembly system as $\mathcal{T} = \{T, (S, (0, 0)), 2\}$, that is, it will consist of tile set T which will contain all 7 of the tile types defined in Figure 2, it will have a seed consisting of a single copy of a tile of type S placed at position $(0, 0)$, and it will be a temperature 2 system (meaning that free tiles need to bind with at least a single strength-2 glue or two individual strength-1 glues on tiles within an existing assembly in order to attach to that assembly).

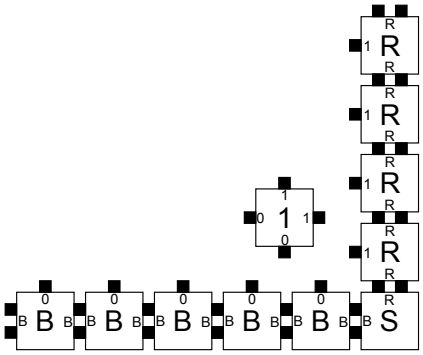


(a) The tile types which form the border of the counter

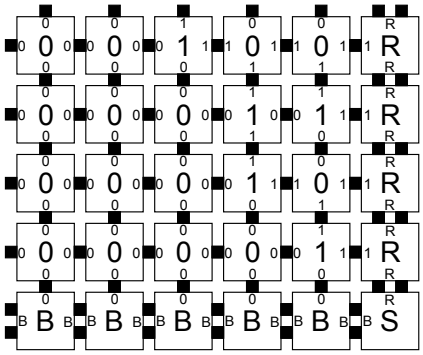
(b) The "rule" tile types which compute and represent the values of the counter

Fig. 2: A tile set which, when seeded with the S tile in a temperature 2 system self-assembles into an infinite binary counter.

Figure 3 shows a small portion of the infinite assembly produced by $\mathcal{T}$. In Figure 3a, the beginning of the formation of the border is shown. Starting from S, border tiles R can attach and form an infinite column upward using their strength-2 glues, and B tiles can do the same to the left. No rule tiles can attach until there are 2 strength-1 bonds correctly positioned for them to bind to. Figure 3a also shows the first rule tile which is about to attach into the corner. In Figure 3b the bottom-right square of width and height 6 of the infinite square assembly is shown. Each horizontal row represents a single binary number in the counter, read from left to right (but which will have an infinite number of leading 0's to the left), and each row represents a binary number exactly one greater than the row immediately beneath it. The computation is performed by the rule tiles which, essentially, receive as "input" a bit from beneath (representing the current value of that column) and a bit from the right (representing the carry bit being brought in). The labels and the northern glues of the rule tiles simply represent the (possibly new) bit to be represented by that column (based on the two inputs), and the western glue represents the resulting carry bit. The computation is possible because of the "cooperation" between two tiles providing input, enforced by the temperature = 2 parameter of the system and the single-strength glues of the rule tiles.



(a) Border tiles can attach to the seed and form arbitrarily long bottom and right borders. Rule tiles can bind only once two "inputs" are available.



(b) A view of the 6×6 square of tiles at the bottom right corner of the assembly produced by the binary counter. Note that the terminal assembly would actually continue infinitely far up and to the left.

Fig. 3: Portions of the assembly formed by the binary counter.

Exercise: performing the XOR operation The goal of this exercise is to gain experience developing a very basic set of tiles which

can perform a simple logical operation on two input bits. Similar to the binary counter, we will assume an infinite assembly bounded on the bottom and right by border tiles. These tiles are shown in Figure 4. Assume that a tile of type S will be placed at location $(0, 0)$ to serve as the seed. Create the set of additional tiles required to take single bit inputs from each of their south and east sides and display as their label the XOR of those two bits, while outputting proper bit values to their north and west.

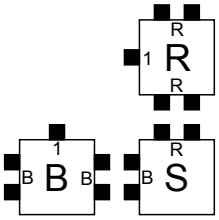


Fig. 4: The border tiles for the XOR exercise

Assuming that all border tiles and tiles with labels equal to 1 are colored red and all other tiles are colored white, what pattern will be displayed on the surface of the assembly created by this system? In other words, what pattern does this system weakly self-assemble?

Exercise: simulating a Turing machine As shown in [44], the aTAM is capable of Turing universal computation. For this exercise, we will explore how, given a particular Turing machine M and a binary string b as input, to design a tile assembly system $\mathcal{T}$ which simulates $M(b)$.

Let M be an extremely basic Turing machine whose state diagram is shown in Figure 5 and $M = (Q, \Sigma, \Gamma, \delta, q_0, q_A, q_R)$, where $Q = (q_0, q_1, q_A, q_R)$ is the set of states, $\Sigma = \{0, 1\}$ is the input alphabet, $\Gamma = \{0, 1, \_\}$ is the tape alphabet, δ is the transition function as shown by Figure 5, q_0 is the start state, q_A is the accept state, and q_R is the reject state.

Goal: Design a tile set which will allow M to be simulated on a specified input.

Tips:

1. Represent M's tape as a row of tiles, with each tile corresponding to a single cell of the tape
2. The cell of the tape which represents the location where M's read head is located will need to be represented by a tile type which includes information about not only the tape cell, but also M's current state
3. Starting with a row of tiles which represents the initial tape and state of M, represent each successive computation step as a row

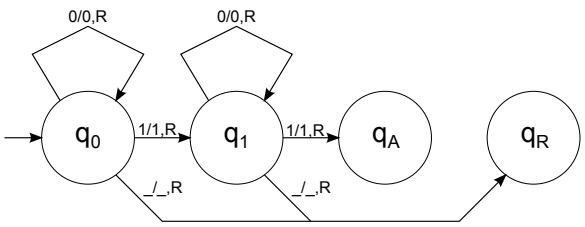


Fig. 5: The state diagram for Turing machine M used in the exercise

of tiles immediately above (which results in a single row of tiles for each computational step and a terminal assembly depicting the entire computational history of $M(b)$, beginning from the bottom and moving upward row by row)

4. Assume that M is designed so that expects to start with its tape head on the leftmost tape cell of the input, the tape is one-way-infinite-to-the-right, and that M will never attempt to move its head left while on the leftmost tape cell

For this exercise, let the input to M be $b = 010010$.

3.3 Survey of aTAM results

Results in the aTAM can often be mapped into two groups: 1. What can or can't self-assemble?, and 2. How hard is it to self-assemble a particular object? Thus, sometimes the interest lies strictly in showing that something is possible or impossible, but often, even though we may know that something is possible, it turns out to be interesting to determine how efficiently it can be done. The most common measure of efficiency is the number of unique tile types required. Finding optimally small tile sets which self-assemble into targeted shapes is of great interest, both theoretically and also for the sake of making potential laboratory implementations more feasible. Another common measure is the scale factor. Oftentimes it is, perhaps counterintuitively, possible to design tile sets with many fewer individual kinds of tiles which can self-assemble a target shape at a blown up scaling factor than it is to self-assemble the same shape without scaling it up. Now we provide a quick overview of a series of results in the aTAM which seek to answer these and other questions.

$n \times n$ squares Since Winfree showed in his thesis [44] that the aTAM is computationally universal, we know that we can algorithmically direct the growth of assemblies. This ability allows for not only the creation of complicated and precise shapes, but also often for them to be created very tile type efficiently (i.e. they require small tile sets - those with few numbers of unique tile types). A benchmark problem for tile-based self-assembly is that of assembling an $n \times n$ square since this requires that the tiles somehow compute the value of n and thus "know when to

stop" at the boundaries. In [38] they showed that binary counters can be used to guide the growth of squares and that thereby it is possible to self-assemble an $n \times n$ square using $O(\log n)$ tile types.
Figure 6 shows a high-level overview of the construction. Essentially, $\log n$ tile types are required so that each bit of (a number related to) the dimension n can be encoded with a unique tile type. The seed is taken to be one of those tile types so that the entire row of them forms attached to the seed. Above that, a fixed-width binary counter (which is composed of the same tile types for all n) begins counting upward from that value until it reaches its maximum possible value (i.e. all 1's), at which point it terminates upward growth. With the vertical bar representing the counter in place, a very basic constant (for all n) set of tiles can be used to "pass a signal" along a diagonal path which is limited by the height (and width) of the counter, and to finally fill in below the diagonal to finish the formation of the square.

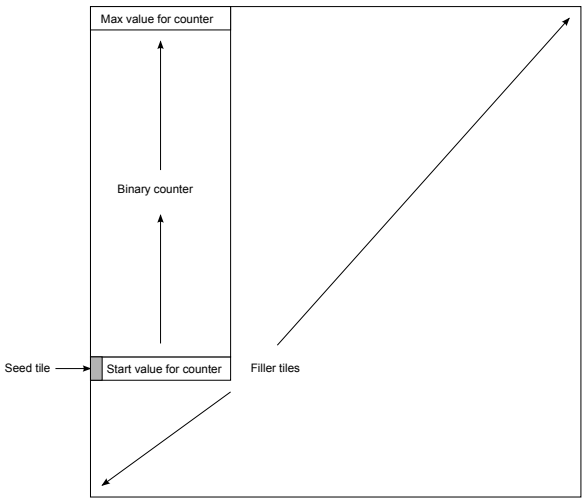


Fig. 6: The high level schematic for building an $n \times n$ square using $O(\log n)$ tile types

In [2], the previous construction for squares was improved to requiring the slightly fewer $O\left(\frac{\log n}{\log \log n}\right)$ tile types, which was also shown to be a matching lower bound (for almost all n).

Finite shapes In order to build any given finite shape, it is trivial to define a tile assembly system which will assemble it: simply create a unique tile type for every point in the shape so that the glue between each tile and each neighbor is unique to that pair in that location. Obvi-

ously, however, this is the worst possible construction in terms of tile type complexity. In [41] it was shown that the tile type complexity for a finite shape S is bounded above and below by the Kolmogorov complexity of the shape, as long as the shape can be scaled up. For the upper bound they provide a construction which uses, in each $c \times c$ square which represents a single point in the original shape (where c is dependent upon S), a Turing machine to read the compressed definition of the shape (from which the tile complexity arises) and then form output sides to that $c \times c$ square which initiate the growth of the necessary squares representing neighboring locations.

Computations Beyond just the relatively simple simulation of a Turing machine on a single input, there have been additional results exploring the power of computation within the aTAM. In [35] it was shown that a set of natural numbers $D \subseteq \mathbb{N}$ is decidable if and only if $D \times \{0\}$ and $D^c \times \{0\}$ weakly self-assemble. That is, the canonical representations of D and the complement of D weakly self-assemble. For $D \times \{0\}$ to weakly self-assemble, at every point along the x-axis such that the value of the x coordinate is contained in D, the tile placed at that location is colored black. All other locations remain either untiled or receive a tile which is not black.
The construction for [35] is a relatively straightforward "stacking" of Turing machine simulations, so that a given Turing machine M which decides the language in question is first simulated on input 0, then immediately above that $M(1)$ is simulated, etc. As each simulation completes, the "answer" of whether or not that input is in the language is propagated via a one-tile-wide path down the side of the previous computations to the x-axis where the appropriately colored tile attaches.
In [25], the more complicated question of whether a similar result applied to computably enumerable languages was answered in the affirmative. It was shown that a set of natural numbers $D \subseteq \mathbb{N}$ is computably enumerable if and only if the set $X_A = \{(f(n), 0) | n \in D\}$ weakly self-assembles (where f is a roughly quadratic function). For that construction, since any Turing machine M used to determine membership in D cannot be guaranteed to halt for non-members, the simple "stacking" construction cannot work. Instead, the construction performs the infinite series of computations side-by-side, spread out along the x-axis (hence the need for f), providing a potentially infinite amount of tape space for each computation while ensuring that none of them collide and a path to the relevant point on the x-axis always remains available for cases in which a black tile must be placed. The space reserved for each computation is achieved by a scheme in which each computation proceeds with each row simply copying the row beneath it for most rows, and then with a frequency half that of the computation to its immediate left, a row performs a new step of the computation. This, combined with a unique and well-defined slope for the assembly representing each computation ensures that the potentially infinite space requirements for every computation can be assured.
Also in [25], it was shown there there exist decidable sets of pairs of integers, i.e. $D \subseteq \mathbb{Z} \times \mathbb{Z}$, which do not weakly self-assemble in the aTAM.

This proof leverages the fact that space is not reusable in tile assembly and that there exist sets for which deciding membership requires too much space to allow each point in an infinite set to be accurately tiled.

Fractals - impossibility and approximation As it has been shown that any finite shape can self-assemble in the aTAM, when looking for shapes which are impossible to self-assemble it is necessary to look at infinite shapes. Due to their complex, aperiodic nature, discrete self-similar fractals have provided an interesting set of shapes to explore.
In [26], it was shown that it is impossible for the discrete Sierpinski triangle to strictly self-assemble in the aTAM (at any temperature). The proof relies on the fact that at each successive stage, as the stages of the fractal structure grow larger, each stage is connected to the rest of the assembly by a single tile. Since there are an infinite number of stages, all of different sizes, it is impossible for the single tiles connecting each of them to the assembly to transmit the information about how large the newly forming stage should be, and thus it is impossible for the fractal to self-assemble. In [34] this proof technique was extended to cover a class of similar fractals. It is conjectured by the author that no discrete self-similar fractal strictly self-assembles in the aTAM, but that remains an open question.
Despite the impossibility of strictly self-assembling the discrete Sierpinski triangle, in [26] it was shown that an approximation of that fractal, which the authors called the *fibered* Sierpinski triangle, does in fact strictly self-assemble. The fibered version is simply a rough visual approximation of the original but with one additional row and column of tiles added to each subsequent stage of the fractal during assembly. Not only does the approximation look similar to the original, it was shown to have the same fractal (or zeta) dimension. In [34], the fibering construction was extended to an entire class of fractals. Along a similar line, in [28] it was shown that a different type of approximation of the Sierpinski triangle strictly self-assembles. This approximation also retains the same approximate appearance and fractal dimension, but instead of "spreading" out successive stages of the fractal with fibering, it utilizes a small portion of each hole in the definition of the shape. In [24], this construction was further extended to an entire class of fractals.

Temperature 1 To this point, all examples, exercises, and results discussed in this paper have been based upon aTAM systems where the temperature is 2. At temperature 2 and above, it is possible to design systems which make use of a feature commonly referred to as *cooperation* in which the correct and prior placement of two tiles in specific relative positions is required before the attachment of a third tile is possible. This cooperative behavior is what is commonly attributed with providing the aTAM with its ability to perform computations, and which (apparently) disappears at temperature = 1. Thus, for aTAM systems whose temperature is 1, it is conjectured that both: 1. Turing universal computation by a deterministic aTAM system is impossible, and 2. any aTAM system which deterministically produces an $n \times n$ square requires a minimum of

$2n-1$ tile types. Partial progress toward the proof of these conjectures was achieved in [18], but the general problem remains open.
Despite the previous conjectures about the aTAM at temperature 1, in [12] it was shown that, by slightly relaxing the requirements, Turing universal computation is in fact possible. Namely, if the assembly is allowed to proceed into the third-dimension, utilizing only 2 planes, or if the computation is allowed to prematurely terminate with some arbitrarily low probability, then a universal Turing machine can be simulated at temperature 1.

Intrinsic universality While an aTAM system can be designed to simulate an arbitrary Turing machine, another interesting question was inspired by the notion of intrinsic universality in cellular automata: Is there a single tile set which can be used to simulate an arbitrary aTAM system? Essentially, if the tiles of this "universal" tile set could be arranged to form a seed structure such that that structure contains an encoding of some other aTAM system, say $\mathcal{T}$, could additional copies of tiles from the universal tile set attach to grow into an assembly which simulates the system $\mathcal{T}$? Of course, the simulation will be a scaled up version of the original system, but it must be the case that every behavior that $\mathcal{T}$ is capable of, the simulating system is also capable of. It turns out that the answer to that question is "yes", as was shown in [17]. In fact, it was shown that there exists a tile set which, when appropriately seeded and at temperature 2, can simulate the behavior of *any* aTAM system at *any* temperature.

Verification Several 'verification problems" (answering the question of whether or not a given system has a specific property) have been studied in relation to the aTAM, and characterized by their complexity. Among them are:

1. Does aTAM system $\mathcal{T}$ uniquely produce a given assembly? This was shown to require time polynomial in the size of the assembly and tile set in [3].
2. Does aTAM system $\mathcal{T}$ uniquely produce a given shape? This was shown to be in co-NP-complete for temperature 1 in [6] and co-NP-complete for temperature 2 in [4].
3. Is a given assembly terminal in aTAM system $\mathcal{T}$? This was shown to require time linear in the size of the assembly and tile set in [3]
4. Given a aTAM system $\mathcal{T}$, does it produce a finite terminal assembly? This was shown to be uncomputable in [6].
5. Given a aTAM system $\mathcal{T}$, does it produce an infinite terminal assembly? This was shown to be uncomputable in [6].

PATS problem and tile set generation In order to produce a surface with a complex template for potentially guiding the attachment of functional materials, an interesting problem in tile-based self-assembly is the Patterned self-Assembly Tile set Synthesis (PATS) problem. The PATS problem is concerned with finding the minimal tile set which will

self-assemble into a given 2-D pattern of colors (where tile types are assumed to be assigned colors). Introduced in [29], in [21] an exhaustive branch-and-bound algorithm was presented which works well for finding exact solutions to patterns of sizes up to 6×6, and approximate solutions for larger patterns. In [27] the previous algorithm was modified to be more efficient (but still requires exponential time).

4 The kinetic Tile Assembly Model (kTAM)

In practice, DNA self-assembly entails a more complicated process than the simple model described by the aTAM, and therefore a different model is required for a realistic simulation of this process. The kinetic Tile Assembly Model(kTAM) [44] is such a model, and considers the reversible nature of self-assembly, factoring in the rates of association and dissociation of basic molecular elements (so-called monomers, or tiles) within the original framework provided by the aTAM. The kTAM describes the dynamics of assembly according to an inclusive set of reversible chemical reactions: A tile can attach to an assembly anywhere that it makes even a weak bond, and any tile can dissociate from the assembly at a rate dependent on the total strength with which it adheres to the assembly.

4.1 Model definition

In the kTAM [20, 44, 46], a monomer tile can be added to the assembly with some association (forward) rate, or removed from the assembly with some dissociation (reverse) rate. Similar to the aTAM, only the singleton tiles are allowed to attach to, and in this case detach from, a seeded assembly. These rates are denoted by r_f and $r_{r,b}$, respectively. At every available site on the perimeter of an assembly (i.e. the *frontier*), every possible monomer tile can associate to the assembly, regardless of whether the monomer is correct or not (i.e. whether or not the glues match). The forward rate depends only on the monomer tile concentration, $[monomer]$:

$$r_f = k_f[monomer] = k_f e^{-G_{mc}} \tag{1}$$

where $G_{mc} > 0$ is the non-dimensional entropic cost of associating to an assembly. In the kTAM, for simplicity it is assumed that tile concentrations remain constant at $[monomer] = e^{-G_{mc}}$. Therefore, since the forward rate constant k_f is a constant, the entire forward rate r_f is also constant.
The reverse rate is dependent upon the binding strength b of the tile to the assembly, and in fact the relationship is exponential:

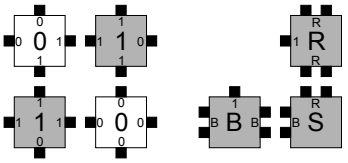
$$r_{r,b} = k_{r,b} = k_f e^{-bG_{se}} \tag{2}$$

where G_{se} is the non-dimensional free energy cost of breaking a single bond and b is the number of "single-strength" bonds the tile has made. The kTAM's equivalent to the aTAM's temperature τ parameter is the ratio of the concentration of the tiles to the strength of their individual

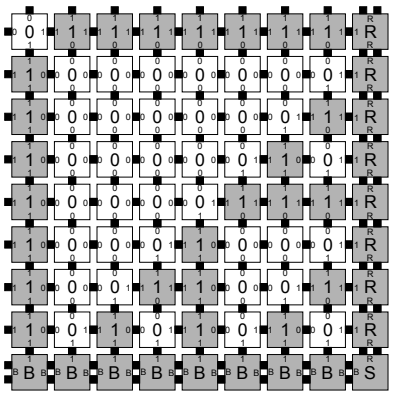
bonds, or G_{mc}/G_{se}. Because the kTAM allows for the binding of tiles whether or not their glues correctly match those on the boundary of a growing assembly, bindings which would be considered errors in the aTAM are possible. By lowering the ratio of G_{mc}/G_{se}, which is intuitively similar to lowering the temperature τ threshold in the aTAM, assembly happens more quickly but is more error prone. If the number of correct bonds that a tile has with an assembly, b, is less than τ, then a tile is more likely to detach than to attach.
Because the kTAM accurately models the behavior of DNA based tile self-assembly in the laboratory, most especially the common types of errors observed, it has provided an excellent foundation for work in error prevention and correction.

4.2 Error types

In order to discuss the types of errors that can occur during self-assembly in the kTAM, we will refer to an example system which is designed to weakly self-assembly the Sierpinski triangle. See Figure 7 for details.



(a) The tile types for weakly self-assembling the Sierpinski triangle

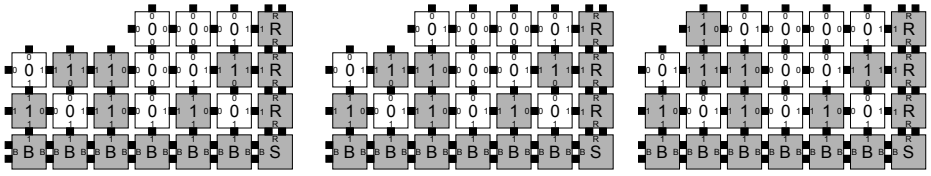


(b) A view of the 9×9 square of tiles at the bottom right corner of the weakly self-assembled Sierpinski triangle. Note that the terminal assembly would actually continue infinitely far up and to the left.

Fig. 7: Details of the Sierpinski triangle example

The errors that occur during assembly can be divided into three general types: 1. *growth errors* (or *mismatch errors*), 2. *facet errors*, and 3. and *nucleation errors* [20]. A growth error, an example of which can be seen in Figure 8, occurs when one or more sides of a tile which binds to an assembly have glues which do not match the adjacent glues (called

glue mismatches). Such a tile may bind with insufficient strength to remain permanently bound, but before it has an opportunity to dissociate, a previously unoccupied neighboring position may be filled by a tile which binds without mismatches, thus resulting in an assembly where every tile has sufficient strength to remain permanently attached despite the mismatch. This essentially "locks" the incorrect tile into place and potentially allows assembly to proceed with an incorrectly placed tile which may cause further deviations from the desired shape or pattern. Somewhat similarly, a facet error also occurs on the edge of a growing assembly. A facet error (see Figure 9 for an example) again occurs when a tile binds with insufficient strength for permanent attachment (but this time with no mismatches), and again is locked into place by a subsequent tile addition. The third type of errors, nucleation errors, occur when tiles aggregate with each other without any attachment to the seed structure, and thus "seed" a new type of assembly.



(a) A partial assembly which is error-free

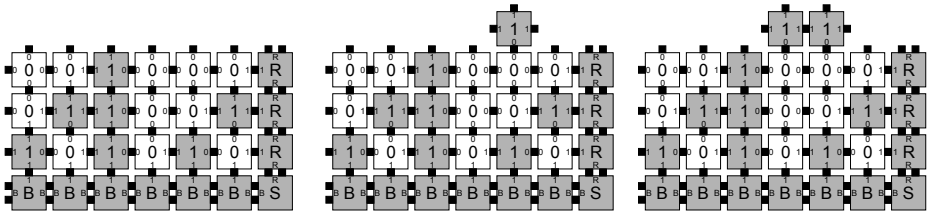
(b) The binding of a tile with one glue match and one mismatch

(c) Before the erroneously attached tile can detach, another tile attaches with 2 matching bonds so that all tiles are now connected by two correctly formed bonds

Fig. 8: Example growth error in the kTAM

4.3 Exercise: error suppression via block replacement

In [46], the authors demonstrated a technique to reduce growth errors which they called *proofreading*. In proofreading, individual tile types are replaced by $n \times n$ blocks of unique tile types such that the perimeter of the $n \times n$ block formed to represent a given tile type from the original set still represents the same glues. (New glues are created for the interior of the block which are specific to the tile types composing each particular block.) However, those original glues are now split into n separate glues. The goal is to force multiple errors to occur before an incorrect $n \times n$ block can fully form, as opposed to the single error which would allow the analogous incorrect tile from the original tile set to bind. They found that by increasing n, it is possible to reduce the growth errors - or alternatively to increase the speed of assembly while maintaining the same error rate.



(a) A partial assembly which is error-free

(b) The binding of a tile via a single glue

(c) Before the erroneously attached tile can detach, another tile attaches with 2 matching bonds so that all tiles are now connected by two correctly formed bonds

Fig. 9: Example facet error in the kTAM

For this exercise, we will construct the 2×2 proofreading tile set for the Sierpinski triangle (shown in its original form in Figure 7a).

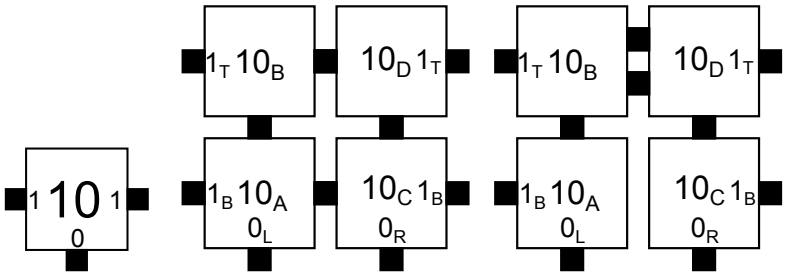
4.4 Survey of kTAM results

We now provide an extremely rough overview of some of results related to the kTAM. Note that there are several laboratory experiments which utilize novel techniques to reduce errors in tile-based self-assembly and to allow for the growth of larger error-free assemblies which are omitted from this discussion.

Facet error handling In [46], the proofreading technique previously discussed was sufficient to reduce growth errors, but was ineffective for handling facet errors. These types of errors were more common in systems "whose growth process[es] intrinsically involve facets", meaning that they frequently require growth to be initiated by extending from a flat surface. In order to reduce these errors, the authors were able to redesign a system used to build an $n \times n$ square by changing the pattern of growth to one which avoids large facets. Specifically, the design used to build the square in Figure 6 was redesigned so that, instead of using a single binary counter growing along one side and then filler tiles which are dependent upon facet growth, two binary counters were used used to form two sides of the square and then filler tiles which use cooperative attachments between those walls. These modifications (along with a few other small changes) were able to greatly reduce the incidence of errors in the growth of squares.

Snaked proofreading In [8], the authors demonstrated a tile set transformation which provided improvements over the previous proofreading technique. In fact, their *snaked proofreading* technique not only

provides substantial improvements in error correction, it also provides for "provably good" assembly time, or specifically that it allows for close to linear assembly time (within logarithmic factors as good as irreversible error-free growth). Snaked proofreading relies on a block replacement scheme similar to the proofreading of [46], but with a different internal bond structure. An example of the difference can be seen in Figure 10. The general technique is to force multiple insufficient attachments to occur and be locked into place before an error can be persisted.



(a) A tile type from the original, unaltered tile set

(b) The block used as a replacement in standard proofreading

(c) The block used as a replacement in snaked proofreading

Fig. 10: A comparison of the block replacement transformations used in standard proofreading and snaked proofreading

It is also further notable that this technique was successfully implemented in the laboratory [10] and the predicted theoretical improvements in controlling facet errors were confirmed.

Self-healing The notion of self-healing, in which a growing assembly is damaged (perhaps by the removal of a group of tiles somewhere in its interior) but then it can correctly re-grow to "heal" the damage without allowing internal errors, was first studied in [45]. The major problem is that many computations are not reversible, but when an assembly receives such damage it is likely to grow on all edges of the hole, and therefore will attempt to grow "backwards" in some areas, causing nondeterministic choices for the inputs to computational steps to frequently result in mistakes.
In [40], it was shown that both proofreading and self-healing properties can be incorporated into tile set transformations which make them robust to both problems simultaneously.

Manipulating tile concentrations to improve assembly In the basic version of the kTAM, it is assumed that not only do the

concentrations of free tiles in solution remain constant during assembly (clearly a simplifying assumption as long as new tiles are not added to the solution), it is also assumed that tiles of all types have the same concentration. In [3] the authors examined the effects of varying the relative concentrations of tile types in order to optimize assembly time (a technique originally introduced in [5]) and provided an algorithm to find the tile type concentrations which approximate the minimum expected assembly time within a $O(\log n)$ factor. In [22] and [9] the authors studied the effects of varying concentrations on both error prevention and assembly time and found that it is possible to improve both. In [9] they showed that the rate of growth errors is minimized by setting the concentration of tiles of type T_i proportional to the square root of the number of times that tiles of type T_i appear in the final assembly (outside of the seed structure). Further, by using those concentrations the expected assembly time is also minimized for constrained systems where the size of the growth frontier (i.e. the number of locations where a tile can attach correctly and with sufficient strength) is limited to 1 at all times. (Note that such systems, although constrained, have been shown to be computationally universal.)

Enhanced tile design While the above (and other) work has successfully demonstrated several techniques for reducing errors that occur during DNA tile-based self-assembly, they have all done so without allowing for the modification of the basic structures of the tiles themselves. However, the simple and static nature of DNA tiles lends itself to the possibility of extension.

In [30], such an extension was proposed. Namely, the authors defined a model in which the "input" glues of tiles are "active" (that is, free to bind to complementary glue strands) when the tiles are freely floating in solution, but their "output" glues are "inactive" (this is, prevented from forming bonds). Only once a tile has associated to an assembly and bound with its input sides are its output sides activated. They presented a theoretical model of such systems and showed that they provide instances of compact (i.e. not requiring scaling factors over the original tile set), error-resilient, and self-healing assembly systems. Furthermore, they provided a possible physical implementation for such systems using DNA polymerase enzymes and strand displacement.

In [20] a similar approach was taken in order to provide for both error-resilience and fast speed of assembly. The *Protected Tile Mechanism* and the *Layered Tile Mechanism*, which utilize stand displacement, were presented. These mechanisms make use of additional DNA strands which "protect", or cover, glues either partially or fully. By balancing the length of the glue strands available for binding on input and output sides at various stages of tile binding, they were able to demonstrate - via simulation - that these mechanisms can in fact improve error rates while maintaining fast assembly.

5 The 2-Handed Assembly Model (2HAM)

5.1 Informal model definition

The 2HAM [11, 13] is a generalization of the aTAM in that it allows for two assemblies, both possibly consisting of more than one tile, to attach to each other. Since we must allow that the assemblies might require translation before they can bind, we define a *supertile* to be the set of all translations of a τ-stable assembly, and speak of the attachment of supertiles to each other, modeling that the assemblies attach, if possible, after appropriate translation. We now give a brief, informal, sketch of the 2HAM.

A *supertile* (a.k.a., *assembly*) is a positioning of tiles on the integer lattice $\mathbb{Z}^2$. Two adjacent tiles in a supertile *interact* if the glues on their abutting sides are equal and have positive strength. Each supertile induces a *binding graph*, a grid graph whose vertices are tiles, with an edge between two tiles if they interact. The supertile 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 supertile is stable if at least energy τ is required to separate the supertile into two parts. A 2HAM *tile assembly system* (TAS) is a pair $\mathcal{T} = (T, \tau)$, where T is a finite tile set and τ is the *temperature*, usually 1 or 2. Given a TAS $\mathcal{T} = (T, \tau)$, a supertile is *producible*, written as $\alpha \in \mathcal{A}[\mathcal{T}]$ if either it is a single tile from T, or it is the τ-stable result of translating two producible assemblies without overlap.[1] A supertile α is *terminal*, written as $\alpha \in \mathcal{A}_{\square}[\mathcal{T}]$ if for every producible supertile β, α and β cannot be τ-stably attached. A TAS is *directed* if it has only one terminal, producible supertile. Given a connected shape $X \subseteq \mathbb{Z}^2$, we say a TAS $\mathcal{T}$ *self-assembles* X if every producible, terminal supertile places tiles exactly on those positions in X (appropriately translated if necessary).

5.2 Formal model definition

We now give a much more formal definition of the 2HAM.

Two assemblies α and β are *disjoint* if $\text{dom } \alpha \cap \text{dom } \beta = \varnothing$. For two assemblies α and β, define the *union* $\alpha \cup \beta$ to be the assembly defined for all $\boldsymbol{x} \in \mathbb{Z}^2$ by $(\alpha \cup \beta)(\boldsymbol{x}) = \alpha(\boldsymbol{x})$ if $\alpha(\boldsymbol{x})$ is defined, and $(\alpha \cup \beta)(\boldsymbol{x}) = \beta(\boldsymbol{x})$ otherwise. Say that this union is *disjoint* if α and β are disjoint.

The *binding graph of* an assembly α is the grid graph $G_\alpha = (V, E)$, where $V = \text{dom } \alpha$, and $\{\boldsymbol{m}, \boldsymbol{n}\} \in E$ if and only if (1) $\boldsymbol{m} - \boldsymbol{n} \in U_2$, (2) $\text{label}_{\alpha(\boldsymbol{m})}(\boldsymbol{n} - \boldsymbol{m}) = \text{label}_{\alpha(\boldsymbol{n})}(\boldsymbol{m} - \boldsymbol{n})$, and (3) $\text{str}_{\alpha(\boldsymbol{m})}(\boldsymbol{n} - \boldsymbol{m}) > 0$. Given $\tau \in \mathbb{N}$, an assembly is *τ-stable* (or simply *stable* if τ is understood from context), if it cannot be broken up into smaller assemblies without breaking bonds of total strength at least τ; i.e., if every cut of G_α has weight at least τ, where the weight of an edge is the strength of the glue it represents. In contrast to the model of Wang tiling, the nonnegativity of the strength function implies that glue mismatches between adjacent tiles

[1] The restriction on overlap is our formalization of the physical mechanism of steric protection.

do not prevent a tile from binding to an assembly, so long as sufficient binding strength is received from the (other) sides of the tile at which the glues match.

For assemblies $\alpha, \beta : \mathbb{Z}^2 \dashrightarrow T$ and $\boldsymbol{u} \in \mathbb{Z}^2$, we write $\alpha + \boldsymbol{u}$ to denote the assembly defined for all $\boldsymbol{x} \in \mathbb{Z}^2$ by $(\alpha + \boldsymbol{u})(\boldsymbol{x}) = \alpha(\boldsymbol{x} - \boldsymbol{u})$, and write $\alpha \simeq \beta$ if there exists $\boldsymbol{u}$ such that $\alpha + \boldsymbol{u} = \beta$; i.e., if α is a translation of β. Define the *supertile* of α to be the set $\tilde{\alpha} = \{ \beta \mid \alpha \simeq \beta \}$. A supertile $\tilde{\alpha}$ is *τ-stable* (or simply *stable*) if all of the assemblies it contains are τ-stable; equivalently, $\tilde{\alpha}$ is stable if it contains a stable assembly, since translation preserves the property of stability. Note also that the notation $|\tilde{\alpha}| \equiv |\alpha|$ is the size of the super tile (i.e., number of tile types in the supertile). is well-defined, since translation preserves cardinality (and note in particular that even though we define $\tilde{\alpha}$ as a set, $|\tilde{\alpha}|$ does not denote the cardinality of this set, which is always $\aleph_0$).

For two supertiles $\tilde{\alpha}$ and $\tilde{\beta}$, and temperature $\tau \in \mathbb{N}$, define the *combination* set $C^\tau_{\tilde{\alpha},\tilde{\beta}}$ to be the set of all supertiles $\tilde{\gamma}$ such that there exist $\alpha \in \tilde{\alpha}$ and $\beta \in \tilde{\beta}$ such that (1) α and β are disjoint (steric protection), (2) $\gamma \equiv \alpha \cup \beta$ is τ-stable, and (3) $\gamma \in \tilde{\gamma}$. That is, $C^\tau_{\tilde{\alpha},\tilde{\beta}}$ is the set of all τ-stable supertiles that can be obtained by attaching $\tilde{\alpha}$ to $\tilde{\beta}$ stably, with $|C^\tau_{\tilde{\alpha},\tilde{\beta}}| > 1$ if there is more than one position at which β could attach stably to α.

It is common with seeded assembly to stipulate an infinite number of copies of each tile, but our definition allows for a finite number of tiles as well. Our definition also allows for the growth of infinite assemblies and finite assemblies to be captured by a single definition, similar to the definitions of [26] for seeded assembly.

Given a set of tiles T, define a *state* S of T to be a multiset of supertiles, or equivalently, S is a function mapping supertiles of T to $\mathbb{N} \cup \{\infty\}$, indicating the multiplicity of each supertile in the state. We therefore write $\tilde{\alpha} \in S$ if and only if $S(\tilde{\alpha}) > 0$.

A *(two-handed) tile assembly system* (*TAS*) is an ordered triple $\mathcal{T} = (T, S, \tau)$, where T is a finite set of tile types, S is the *initial state*, and $\tau \in \mathbb{N}$ is the temperature. If not stated otherwise, assume that the initial state S is defined $S(\tilde{\alpha}) = \infty$ for all supertiles $\tilde{\alpha}$ such that $|\tilde{\alpha}| = 1$, and $S(\tilde{\beta}) = 0$ for all other supertiles $\tilde{\beta}$. That is, S is the state consisting of a countably infinite number of copies of each individual tile type from T, and no other supertiles. In such a case we write $\mathcal{T} = (T, \tau)$ to indicate that $\mathcal{T}$ uses the default initial state.

Given a TAS $\mathcal{T} = (T, S, \tau)$, define an *assembly sequence* of $\mathcal{T}$ to be a sequence of states $\boldsymbol{S} = (S_i \mid 0 \leq i < k)$ (where $k = \infty$ if $\boldsymbol{S}$ is an infinite assembly sequence), and S_{i+1} is constrained based on S_i in the following way: There exist supertiles $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}$ such that (1) $\tilde{\gamma} \in C^\tau_{\tilde{\alpha},\tilde{\beta}}$, (2) $S_{i+1}(\tilde{\gamma}) = S_i(\tilde{\gamma}) + 1$,[2] (3) if $\tilde{\alpha} \neq \tilde{\beta}$, then $S_{i+1}(\tilde{\alpha}) = S_i(\tilde{\alpha}) - 1$, $S_{i+1}(\tilde{\beta}) = S_i(\tilde{\beta}) - 1$, otherwise if $\tilde{\alpha} = \tilde{\beta}$, then $S_{i+1}(\tilde{\alpha}) = S_i(\tilde{\alpha}) - 2$, and (4) $S_{i+1}(\tilde{\omega}) = S_i(\tilde{\omega})$ for all $\tilde{\omega} \notin \{\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}\}$. That is, S_{i+1} is obtained from S_i by picking two supertiles from S_i that can attach to each other, and attaching them, thereby decreasing the count of the two reactant

[2] with the convention that $\infty = \infty + 1 = \infty - 1$

supertiles and increasing the count of the product supertile. If $S_0 = S$, we say that $\boldsymbol{S}$ is *nascent*.

Given an assembly sequence $\boldsymbol{S} = (S_i \mid 0 \leq i < k)$ of $\mathcal{T} = (T, S, \tau)$ and a supertile $\tilde{\gamma} \in S_i$ for some i, define the *predecessors* of $\tilde{\gamma}$ in $\boldsymbol{S}$ to be the multiset $\text{pred}_{\boldsymbol{S}}(\tilde{\gamma}) = \{\tilde{\alpha}, \tilde{\beta}\}$ if $\tilde{\alpha}, \tilde{\beta} \in S_{i-1}$ and $\tilde{\alpha}$ and $\tilde{\beta}$ attached to create $\tilde{\gamma}$ at step i of the assembly sequence, and define $\text{pred}_{\boldsymbol{S}}(\tilde{\gamma}) = \{\tilde{\gamma}\}$ otherwise. Define the *successor* of $\tilde{\gamma}$ in $\boldsymbol{S}$ to be $\text{succ}_{\boldsymbol{S}}(\tilde{\gamma}) = \tilde{\alpha}$ if $\tilde{\gamma}$ is a predecessor of $\tilde{\alpha}$ in $\boldsymbol{S}$, and define $\text{succ}_{\boldsymbol{S}}(\tilde{\gamma}) = \tilde{\gamma}$ otherwise. A sequence of supertiles $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_i \mid 0 \leq i < k)$ is a *supertile assembly sequence* of $\mathcal{T}$ if there is an assembly sequence $\boldsymbol{S} = (S_i \mid 0 \leq i < k)$ of $\mathcal{T}$ such that, for all $1 \leq i < k$, $\text{succ}_{\boldsymbol{S}}(\tilde{\alpha}_{i-1}) = \tilde{\alpha}_i$, and $\tilde{\boldsymbol{\alpha}}$ is *nascent* if $\boldsymbol{S}$ is nascent.

The *result* of a supertile assembly sequence $\tilde{\boldsymbol{\alpha}}$ is the unique supertile $\text{res}(\tilde{\boldsymbol{\alpha}})$ such that there exist an assembly $\alpha \in \text{res}(\tilde{\boldsymbol{\alpha}})$ and, for each $0 \leq i < k$, assemblies $\alpha_i \in \tilde{\alpha}_i$ such that $\text{dom}\ \alpha = \bigcup_{0 \leq i < k} \text{dom}\ \alpha_i$ and, for each $0 \leq i < k$, $\alpha_i \sqsubseteq \alpha$. For all supertiles $\tilde{\alpha}, \tilde{\beta}$, we write $\tilde{\alpha} \to_{\mathcal{T}} \tilde{\beta}$ (or $\tilde{\alpha} \to \tilde{\beta}$ when $\mathcal{T}$ is clear from context) to denote that there is a supertile assembly sequence $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_i \mid 0 \leq i < k)$ such that $\tilde{\alpha}_0 = \tilde{\alpha}$ and $\text{res}(\tilde{\boldsymbol{\alpha}}) = \tilde{\beta}$. It can be shown using the techniques of [37] for seeded systems that for all two-handed tile assembly systems $\mathcal{T}$ supplying an infinite number of each tile type, $\to_{\mathcal{T}}$ is a transitive, reflexive relation on supertiles of $\mathcal{T}$. We write $\tilde{\alpha} \to^1_{\mathcal{T}} \tilde{\beta}$ ($\tilde{\alpha} \to^1 \tilde{\beta}$) to denote an assembly sequence of length 1 from $\tilde{\alpha}$ to $\tilde{\beta}$ and $\tilde{\alpha} \to^{\leq 1}_{\mathcal{T}} \tilde{\beta}$ ($\tilde{\alpha} \to^{\leq 1} \tilde{\beta}$) to denote an assembly sequence of length 1 from $\tilde{\alpha}$ to $\tilde{\beta}$ if $\tilde{\alpha} \neq \tilde{\beta}$ and an assembly sequence of length 0 otherwise.

A supertile $\tilde{\alpha}$ is *producible*, and we write $\tilde{\alpha} \in \mathcal{A}[\mathcal{T}]$, if it is the result of a nascent supertile assembly sequence. A supertile $\tilde{\alpha}$ is *terminal* if, for all producible supertiles $\tilde{\beta}$, $C^{\tau}_{\tilde{\alpha},\tilde{\beta}} = \varnothing$.[3] Define $\mathcal{A}_{\square}[\mathcal{T}] \subseteq \mathcal{A}[\mathcal{T}]$ to be the set of terminal and producible supertiles of $\mathcal{T}$. $\mathcal{T}$ is *directed* (a.k.a., *deterministic*, *confluent*) if $|\mathcal{A}_{\square}[\mathcal{T}]| = 1$.

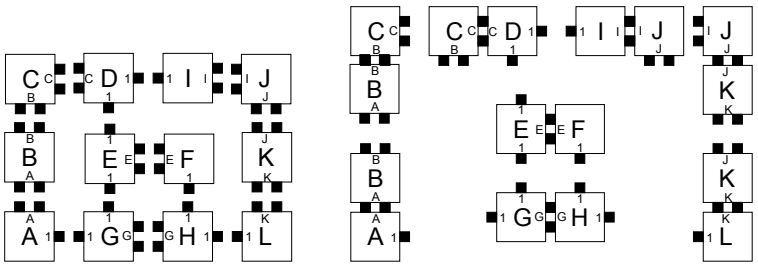
Let $X \subseteq \mathbb{Z}^2$ be a shape. We say X *self-assembles* in $\mathcal{T}$ if, for each $\tilde{\alpha} \in \mathcal{A}_{\square}[\mathcal{T}]$, there exists $\alpha \in \tilde{\alpha}$ such that $\text{dom}\ \alpha = X$; i.e., $\mathcal{T}$ uniquely assembles into the shape X.

5.3 Examples and exercises

In this section we provide an example of a simple 2HAM system and show exactly what assemblies are producible within it, and then give an exercise for developing a 2HAM system.

[3] Note that a supertile $\tilde{\alpha}$ could be non-terminal in the sense that there is a producible supertile $\tilde{\beta}$ such that $C^{\tau}_{\tilde{\alpha},\tilde{\beta}} \neq \varnothing$, yet it may not be possible to produce $\tilde{\alpha}$ and $\tilde{\beta}$ simultaneously if some tile types are given finite initial counts, implying that $\tilde{\alpha}$ cannot be "grown" despite being non-terminal. If the count of each tile type in the initial state is ∞, then all producible supertiles are producible from any state, and the concept of terminal becomes synonymous with "not able to grow", since it would always be possible to use the abundant supply of tiles to assemble $\tilde{\beta}$ alongside $\tilde{\alpha}$ and then attach them.

An example 2HAM system Let $\mathcal{T} = (T, 2)$ be a 2HAM system where T is defined as the tile types in Figure 11a. Figures 11a-12c show the complete set of 29 supertiles which make up $\mathcal{A}[\mathcal{T}]$, and Figure 12c shows the single member of $\mathcal{A}_{\square}[\mathcal{T}]$. The producible supertiles are broken into groups to show the earliest step of combinations during which they can appear, although for some there are multiple paths of combinations which can form them. (We don't show duplicate copies.) Furthermore, recall from the definition of the model that all producible supertiles are available at every step, so for example a supertile produced in step 2 may combine with one produced in step 1 to create a new supertile in step 3. Also note that the use of "steps" is merely a convenience for discussing this example, but typically the sets $\mathcal{A}[\mathcal{T}]$ and $\mathcal{A}_{\square}[\mathcal{T}]$ are simply defined as those supertiles producible in the limit.



(a) The tile set (a.k.a. singleton tiles) for the 2HAM example system

(b) The new supertiles producible after one step of combinations

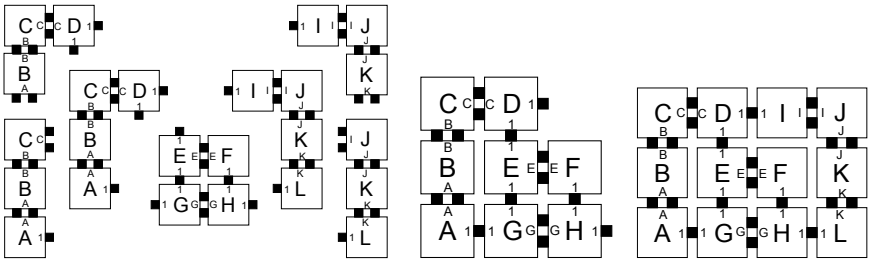
Fig. 11: An example 2HAM system and some producible assemblies

Building frames Define a *frame* as a hollow $m \times n$ rectangle where $m, n > 2$, i.e. a rectangle with both dimensions at least size 3 and tiles only on its perimeter. (See Figure 13 for an example.) Create a tile set T such that $\mathcal{T} = (T, 2)$ and $\mathcal{A}_{\square}[\mathcal{T}]$ is exactly the infinite set of frames.

5.4 Survey of 2HAM results

We now provide a brief, incomplete sketch of some results in the 2HAM.

Simulation of the aTAM The aTAM assumes a controlled, well-defined origin for the initiation of all assemblies, while the 2HAM allows for "spontaneous" nucleation caused by any two producible assemblies (including singleton tiles) which can bind with sufficient strength. Given this much greater level of freedom, the question of whether or not that could be constrained and forced to behave in a way similar to the aTAM



(a) The new supertiles producible after the second step of combinations

(b) The only new supertile producible after the third step of combinations

(c) The only new supertile producible after the fourth step of combinations, and which is the unique terminal assembly of the system

Fig. 12: Continuation of the example 2HAM system's producible assemblies

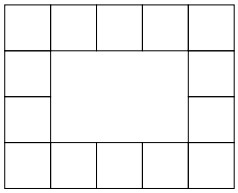


Fig. 13: An example 5×4 frame

was asked in [6]. The answer was "yes", and in fact in [6] a construction was presented which, given an arbitrary aTAM system $\mathcal{T}$, provides for a way to construct a 2HAM system $\mathcal{S}$ which can faithfully simulate $\mathcal{T}$. The cost is a mere constant scaling factor of 5. The general technique is to allow $\mathcal{S}$ to form 5×5 blocks which represent the tiles in $\mathcal{T}$ but in a very constrained way so that the blocks can only fully form and present their output glues once they've attached to a growing assembly which contains a seed block (and therefore they can't spontaneously combine away from the "seeded" assembly).

Verification problems Given that the 2HAM allows for a greater variety of behaviors than the aTAM, and in fact in some sense for the transmission of information over arbitrary distances (by the placements of glues and general geometric shapes of arbitrarily large supertiles which are combining), it shouldn't be surprising that several "verification problems" (answering the question of whether or not a given system has a specific property) are more difficult for the 2HAM. Several verification

problems have been characterized in terms of their complexity, some of which include:

1. Does 2HAM system $\mathcal{T}$ uniquely produce a given assembly? This was shown to be co-NP-complete for 3D temperature 2 systems in the 2HAM in [6]. (But note that it is solvable in polynomial time in the aTAM under the same parameters!)
2. Does 2HAM system $\mathcal{T}$ uniquely produce a given shape? This was shown to be in co-NP for temperature 1 and co-NP-complete for temperature 2 in [4].
3. Is a given assembly terminal in 2HAM system $\mathcal{T}$? In [6] this was shown to be uncomputable for temperature 2 systems in the 2HAM (while it is computable in polynomial time in the aTAM [3], and also for the 2HAM at temperature 1 [6].)
4. Given a 2HAM system $\mathcal{T}$, does it produce a finite terminal assembly? This was shown to be uncomputable in [6].
5. Given a 2HAM system $\mathcal{T}$, does it produce an infinite terminal assembly? This was shown to be uncomputable for temperature 2 2HAM systems in [6].

Impossibility and efficiency comparisons with the aTAM Given that the 2HAM can simulate the aTAM (and that the converse is not true), it is somewhat surprising that in [6] it was shown that there is a simple class of shapes (so-called *loops*) which can be assembled with slightly greater tile type efficiency in the aTAM at temperature 1 than in the 2HAM at temperature 1. (However, this separation disappears at temperature 2.) Nonetheless, in [6] it was also shown that there are shapes called *staircases* which can self-assemble in the 2HAM using roughly n tile types, while the aTAM requires a number exponential in n (and this can in fact be extended to the busy beaver function, $BB(n)$). In terms of impossibility, it was shown that there is a class of infinite shapes which self-assembles in the aTAM but not the 2HAM, and also a class of shapes shape which can self-assemble (in a weaker sense) in the 2HAM but not in the aTAM.

Speed of assembly Since the 2HAM allows for assemblies to begin forming in parallel and then to combine in pairs, it would seem that perhaps this would allow for sublinear assembly times. However, in [7] they developed a physically realistic timing model for the 2HAM (referred to there as the *Hierarchical aTAM*) and showed that it is impossible to build shapes of diameter n in time less than $\Omega(n)$ in deterministic systems. However, they then exhibited a nondeterministic system which can assemble an $n \times n'$ rectangle (where $n > n'$) in time $O(n^{4/5} \log n)$, breaking the linear-time lower bound (which applies not only to deterministic 2HAM systems, but also to seeded aTAM systems).

6 Newer Models

In this section, we provide extremely high-level descriptions of a variety of newer models that have been derived from the TAM.

6.1 Temperature programming

In the standard aTAM, the "program" that is being executed during self-assembly can be thought of as being specified by the specific tile types of the system. It is the information encoded in the glues that direct the behavior of the system and guide assembly. Introduced in [4], the multiple temperature model, or *temperature programming*, is a variant of the seeded aTAM which allows for the temperature of the system to be changed (raised or lowered) during the assembly process and at well defined points. More specifically, a series of temperature transitions, along with the tile set, seed, and initial temperature, are specified. Assembly progresses from the seed until it is terminal. At that point, the first temperature transition is made and assembly continues until it is terminal. If another temperature transition has been specified it is made and assembly once again continues, and so on until assembly is terminal and no additional temperature transitions have been specified. The addition of a series of temperature transitions as input turns out to be a powerful tool, and, among other results for this model, in [42] it was shown that there exist systems using one of two constant tile sets that can self-assemble scaled-up versions of arbitrary shapes. One system uses a larger scaling factor dependent upon the shape but a "Kolmogorov-optimum" temperature sequence, while the other uses a small, constant scaling factor but a temperature sequence proportional to the number of points in the shape. In [42] it was also shown that there exists no single tile set which can self-assemble an arbitrary shape in this model without scaling.

6.2 Concentration programming

Somewhat akin to the multiple temperature model, *tile concentration programming*, introduced in [5], allows for the inclusion of additional information as input to a tile assembly system. In this model, that information is provided as the relative concentrations of the various tile types. As mentioned in 4.4, this tool has been used for reducing both assembly time and the frequency of errors in the kTAM. It has also been used in a variant of the aTAM to provide nondeterministic "competitions" between tiles of different types for binding at specified locations. The results of these competitions can be used by the system to sample the relative concentrations of the tile types and thus "read" the input information that provides. In a series of results from [5] to [23] to [15], it was shown how to use this information to build shapes such as squares. Most recently, in [15] it was shown how to combine tile concentration programming with a constant tile set to form any $n \times n$ square with high probability (for sufficiently large n), and also how to self-assemble arbitrary scaled shapes using a constant tile set and tile type concentrations dependent upon the definition of the shape.

6.3 Repulsive glues

In the aTAM, all pairs of glues interact with either a positive (i.e. attractive) force when the glues match, or no force at all when the glues

do not match. However, in natural systems there is also another option: a negative (i.e. repulsive) force. For instance, two objects with the same electric charge (or two magnets of opposite orientation) will repel each other. Several variations of models allowing so-called *negative glues* have been defined, along with a series of related results. See [16, 33, 36] for examples.

6.4 Staged self-assembly

Self-assembly in the aTAM is considered a "one pot" reaction, meaning that all assembly for a given system occurs in one test tube. Furthermore, during the entire assembly process all tile types are present. In [13] the authors defined a model in which different subsets of tile types and currently produced assemblies can be placed into distinct test tubes, or bins, for portions of the assembly process. Once each bin has reached a terminal state, it is possible to combine or separate the contents of bins and individual tile types into new bins, and perform the next *stage* of assembly. This increases the resources required for a self-assembling system, but provides additional input in the form of the staging algorithm (the definition of the series of stages) and dramatically increases in the power of such systems. For instance, in [13] they were able to demonstrate that a constant tile set can be used to self-assemble arbitrary shapes - with no scaling! This construction requires a number of bins and stages dependent on the particular shape, and they presented a variety of constructions which exhibited tradeoffs between the number of tile types, number of bins, number of stages, and scaling factor.

Staged assembly with RNase As an extension to staged self-assembly 6.4, in [1] they introduced the ability to create tile types out of two different materials (e.g. DNA and RNA) and then to allow for the dissolution of one type (e.g. RNA tiles) at specified points during the assembly by the addition of an enzyme (e.g. an RNase enzyme). This additional power allowed them to perform replication of input shapes. In further work, in [14] it was shown how to self-assemble arbitrary shapes using an asymptotically optimal number of tile types, a scaling factor related to the log of the shape's size, and a constant number of stages.

6.5 Geometrically complex tiles

Work in the aTAM is generally done with the assumption of a "diagonal" glue function, which means that the function that maps the strength of interaction between pairs of glues returns a 0 for all pairs of glues where both are not the same glue type, and a positive number for pairs of glues of matching type. Given such a glue function, which is the standard, as previously mentioned the lower bound on the unique number of tile types which can self-assemble an $n \times n$ square is $O\left(\frac{\log n}{\log \log n}\right)$. However, for a non-diagonal glue function, which is one that allows interactions between each glue type and any subset of other glue types, that lower

bound falls to $\sqrt{\log n}$. In order to provide a potentially realistic means of implementing non-diagonal glue functions, the Geometric Tile Assembly Model (GTAM) was introduced in [19], and a series of constructions in the GTAM were presented which: 1. self-assemble an $n \times n$ square in the optimal $O(\sqrt{\log n})$ tile types and at temperature 1, 2. simulate a computationally universal class of temperature 2 aTAM constructions at temperature 1, and 3. in a 2-handed version of the GTAM and allowing 4 planes to be used in the third dimension, self-assembles an $n \times n$ square using only $O(\log(\log n))$ tile types.

6.6 Signal passing tiles

In the previously discussed models (other than those in Section 4.4), the tiles are static objects which do not change in structure or function upon binding. To study a more "active" model, in [31] the Signal passing Tile Assembly Model (STAM), which is based on the 2HAM, was introduced. In the STAM, tiles are allowed to have possibly multiple glues on each side. At any point in time each glue can be in one of three states: 1. "`latent`" (inactive and has never been active), 2. "`on`" (active, available to bind), and 3. "`off`" (has been deactivated). A tile's glues can initially begin as either `latent` or `on`. Only glues which are `on` are able to bind, and when a glue binds it is possible for it to signal any subset of glues on the same tile to perform one of the following transitions: 1. `latent` $\rightarrow$ `on`, 2. `latent` $\rightarrow$ `off`, or 3. `on` $\rightarrow$ `off`. The STAM is highly asynchronous, so there is no guarantee about when a signal will be acted upon, only that it will happen at some point in the future. It is important to note that each tile has a constant number of glues and thus signals that it can initiate and react to.
Complexity analysis of STAM systems includes the maximum number of glues that appear on the face of any tile in a given system (called the *signal complexity*), and in [31] the authors demonstrated a construction which is able to self-assemble a $1 \times n$ line with a constant number of tile types and signal complexity $O(\log n)$. They also presented a construction which is able to simulate a Turing machine without making a copy of the entire row representing the tape at each step, but which instead uses only a constant number of new tiles per step. Their final construction is the first known of any model which can strictly self-assemble a discrete self-similar fractal, namely the Sierpinski triangle (which is provably impossible in models such as the aTAM and 2HAM).

References

1. Zachary Abel, Nadia Benbernou, Mirela Damian, Erik Demaine, Martin Demaine, Robin Flatland, Scott Kominers, and Robert Schweller, *Shape replication through self-assembly and RNase enzymes*, SODA 2010: Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms (Austin, Texas), Society for Industrial and Applied Mathematics, 2010.

2. Leonard Adleman, Qi Cheng, Ashish Goel, and Ming-Deh Huang, *Running time and program size for self-assembled squares*, Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (Hersonissos, Greece), 2001, pp. 740–748.
3. Leonard M. Adleman, Qi Cheng, Ashish Goel, Ming-Deh A. Huang, David Kempe, Pablo Moisset de Espanés, and Paul W. K. Rothemund, *Combinatorial optimization problems in self-assembly*, Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing, 2002, pp. 23–32.
4. Gagan Aggarwal, Michael H. Goldwasser, Ming-Yang Kao, and Robert T. Schweller, *Complexities for generalized models of self-assembly*, Proceedings of ACM-SIAM Symposium on Discrete Algorithms, 2004.
5. Florent Becker, Ivan Rapaport, and Eric Rémila, *Self-assembling classes of shapes with a minimum number of tiles, and in optimal time*, Foundations of Software Technology and Theoretical Computer Science (FSTTCS), 2006, pp. 45–56.
6. 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)*, Tech. Report 1201.1650, Computing Research Repository, 2012.
7. Ho-Lin Chen and David Doty, *Parallelism and time in hierarchical self-assembly*, SODA 2012: Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2012, pp. 1163–1182.
8. Ho-Lin Chen and Ashish Goel, *Error free self-assembly using error prone tiles*, Proceedings of the 10th International Meeting on DNA Based Computers, 2004, pp. 274–283.
9. Ho-Lin Chen and Ming-Yang Kao, *Optimizing tile concentrations to minimize errors and time for dna tile self-assembly systems.*, DNA (Yasubumi Sakakibara and Yongli Mi, eds.), Lecture Notes in Computer Science, vol. 6518, Springer, 2010, pp. 13–24.
10. Ho-Lin Chen, Rebecca Schulman, Ashish Goel, and Erik Winfree, *Reducing facet nucleation during algorithmic self-assembly*, Nano Letters **7** (2007), no. 9, 2913–2919.
11. Qi Cheng, Gagan Aggarwal, Michael H. Goldwasser, Ming-Yang Kao, Robert T. Schweller, and Pablo Moisset de Espanés, *Complexities for generalized models of self-assembly*, SIAM Journal on Computing **34** (2005), 1493–1515.
12. Matthew Cook, Yunhui Fu, and Robert T. Schweller, *Temperature 1 self-assembly: Deterministic assembly in 3D and probabilistic assembly in 2D*, SODA 2011: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2011.
13. 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.
14. Erik D. Demaine, Matthew J. Patitz, Robert T. Schweller, and Scott M. Summers, *Self-assembly of arbitrary shapes using rnase*

enzymes: Meeting the kolmogorov bound with small scale factor (extended abstract)., STACS (Thomas Schwentick and Christoph Drr, eds.), LIPIcs, vol. 9, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011, pp. 201–212.

15. David Doty, *Randomized self-assembly for exact shapes*, SIAM Journal on Computing **39** (2010), no. 8, 3521–3552.
16. David Doty, Lila Kari, and Benoît Masson, *Negative interactions in irreversible self-assembly*, Algorithmica, to appear. Preliminary version appeared in DNA 2010.
17. David Doty, Jack H. Lutz, Matthew J. Patitz, Robert T. Schweller, Scott M. Summers, and Damien Woods, *The tile assembly model is intrinsically universal*, Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, 2012, to appear.
18. David Doty, Matthew J. Patitz, and Scott M. Summers, *Limitations of self-assembly at temperature 1*, Theoretical Computer Science **412** (2011), 145–158.
19. Bin Fu, Matthew J. Patitz, Robert T. Schweller, and Robert Sheline, *Self-assembly with geometric tiles*, Proceedings of the 39th International Colloquium on Automata, Languages and Programming, ICALP, 2012, to appear.
20. Kenichi Fujibayashi, David Yu Zhang, Erik Winfree, and Satoshi Murata, *Error suppression mechanisms for dna tile self-assembly and their simulation*, Natural Computing: an international journal **8** (2009), no. 3, 589–612.
21. Mika Göös and Pekka Orponen, *Synthesizing minimal tile sets for patterned dna self-assembly*, DNA (Yasubumi Sakakibara and Yongli Mi, eds.), Lecture Notes in Computer Science, vol. 6518, Springer, 2010, pp. 71–82.
22. Byunghyun Jang, Yong-Bin Kim, and Fabrizio Lombardi, *Error tolerance of dna self-assembly by monomer concentration control*, Defect and Fault-Tolerance in VLSI Systems, IEEE International Symposium on **0** (2006), 89–97.
23. Ming-Yang Kao and Robert T. Schweller, *Randomized self-assembly for approximate shapes.*, ICALP (1) (Luca Aceto, Ivan Damgrd, Leslie Ann Goldberg, Magns M. Halldrsson, Anna Inglfsdttir, and Igor Walukiewicz, eds.), Lecture Notes in Computer Science, vol. 5125, Springer, 2008, pp. 370–384.
24. Steven M. Kautz and Brad Shutters, *Self-assembling rulers for approximating generalized sierpinski carpets*, COCOON (Bin Fu and Ding-Zhu Du, eds.), Lecture Notes in Computer Science, vol. 6842, Springer, 2011, pp. 284–296.
25. James I. Lathrop, Jack H. Lutz, Matthew J. Patitz, and Scott M. Summers, *Computability and complexity in self-assembly*, Theory Comput. Syst. **48** (2011), no. 3, 617–647.
26. James I. Lathrop, Jack H. Lutz, and Scott M. Summers, *Strict self-assembly of discrete Sierpinski triangles*, Theoretical Computer Science **410** (2009), 384–405.
27. Tuomo Lempiäinen, Eugen Czeizler, and Pekka Orponen, *Synthesizing small and reliable tile sets for patterned dna self-assembly*,

Proceedings of the 17th international conference on DNA computing and molecular programming (Berlin, Heidelberg), DNA'11, Springer-Verlag, 2011, pp. 145–159.
28. Jack H. Lutz and Brad Shutters, *Approximate self-assembly of the sierpinski triangle*, Theory Comput. Syst. **51** (2012), no. 3, 372–400.
29. Xiaojun Ma and Fabrizio Lombardi, *Synthesis of tile sets for dna self-assembly*, IEEE Trans. on CAD of Integrated Circuits and Systems **27** (2008), no. 5, 963–967.
30. Urmi Majumder, Thomas H. Labean, and John H. Reif, *Activatable tiles: Compact, robust programmable assembly and other applications*, in DNA Computing: DNA13 (edited by Max Garzon and Hao Yan), Springer-Verlag Lecture Notes for Computer Science (LNCS, Springer, 2007, pp. 15–25.
31. Jennifer E. Padilla, Matthew J. Patitz, Raul Pena, Robert T. Schweller, Nadrian C. Seeman, Robert Sheline, Scott M. Summers, and Xingsi Zhong, *Asynchronous signal passing for tile self-assembly: Fuel efficient computation and efficient assembly of shapes*, Tech. Report 1202.5012, Computing Research Repository, 2012.
32. Matthew J. Patitz, *Simulation of self-assembly in the abstract tile assembly model with ISU TAS*, 6th Annual Conference on Foundations of Nanoscience: Self-Assembled Architectures and Devices (Snowbird, Utah, USA, April 20-24 2009)., 2009.
33. Matthew J. Patitz, Robert T. Schweller, and Scott M. Summers, *Exact shapes and turing universality at temperature 1 with a single negative glue*, Proceedings of the 17th international conference on DNA computing and molecular programming (Berlin, Heidelberg), DNA'11, Springer-Verlag, 2011, pp. 175–189.
34. Matthew J. Patitz and Scott M. Summers, *Self-assembly of discrete self-similar fractals*, Natural Computing **1** (2010), 135–172.
35. _____, *Self-assembly of decidable sets*, Natural Computing **10** (2011), no. 2, 853–877.
36. John Reif, Sudheer Sahu, and Peng Yin, *Complexity of graph self-assembly in accretive systems and self-destructible systems*, DNA Computing (Alessandra Carbone and Niles Pierce, eds.), Lecture Notes in Computer Science, vol. 3892, Springer Berlin / Heidelberg, 2006, pp. 257–274.
37. Paul W. K. Rothemund, *Theory and experiments in algorithmic self-assembly*, Ph.D. thesis, University of Southern California, December 2001.
38. Paul W. K. Rothemund and Erik Winfree, *The program-size complexity of self-assembled squares (extended abstract)*, STOC '00: Proceedings of the thirty-second annual ACM Symposium on Theory of Computing (Portland, Oregon, United States), ACM, 2000, pp. 459–468.
39. Nadrian C. Seeman, *Nucleic-acid junctions and lattices*, Journal of Theoretical Biology **99** (1982), 237–247.
40. David Soloveichik, Matthew Cook, and Erik Winfree, *Combining self-healing and proofreading in self-assembly.*, Natural Computing **7** (2008), no. 2, 203–218.
41. David Soloveichik and Erik Winfree, *Complexity of self-assembled shapes*, SIAM Journal on Computing **36** (2007), no. 6, 1544–1569.

42. Scott M. Summers, *Reducing tile complexity for the self-assembly of scaled shapes through temperature programming*, Algorithmica **63** (2012), no. 1-2, 117–136.
43. Hao Wang, *Dominoes and the AEA case of the decision problem*, Proceedings of the Symposium on Mathematical Theory of Automata (New York, 1962), Polytechnic Press of Polytechnic Inst. of Brooklyn, Brooklyn, N.Y., 1963, pp. 23–55.
44. Erik Winfree, *Algorithmic self-assembly of DNA*, Ph.D. thesis, California Institute of Technology, June 1998.
45. _____, *Self-healing tile sets*, Nanotechnology: Science and Computation (Junghuei Chen, Natasa Jonoska, and Grzegorz Rozenberg, eds.), Natural Computing Series, Springer, 2006, pp. 55–78.
46. Erik Winfree and Renat Bekbolatov, *Proofreading tile sets: Error correction for algorithmic self-assembly.*, DNA (Junghuei Chen and John H. Reif, eds.), Lecture Notes in Computer Science, vol. 2943, Springer, 2003, pp. 126–144.