Accelerating Self-Assembly of Crisscross Slat Systems

David Doty
University of California–Davis, CA, USA

Hunter Fleming
University of Arkansas, Fayetteville, AR, USA

Daniel Hader
University of Arkansas, Fayetteville, AR, USA

Matthew J. Patitz
University of Arkansas, Fayetteville, AR, USA

Lukas A. Vaughan
University of Arkansas, Fayetteville, AR, USA

Abstract

We present an abstract model of self-assembly of systems composed of "crisscross slats", which have been experimentally implemented as a single-stranded piece of DNA [21] or as a complete DNA origami structure [28]. We then introduce a more physically realistic "kinetic" model and show how important constants in the model were derived and tuned, and compare simulation-based results to experimental results [21, 28]. Using these models, we show how we can apply optimizations to designs of slat systems in order to lower the numbers of unique slat types required to build target structures. In general, we apply two types of techniques to achieve greatly reduced numbers of slat types. Similar to the experimental work implementing DNA origami-based slats, in our designs the slats oriented in horizontal and vertical directions are each restricted to their own plane and sets of them overlap each other in square regions which we refer to as *macrotiles*. Our first technique extends their previous work of reusing slat types within macrotiles and requires analyses of binding domain patterns to determine the potential for errors consisting of incorrect slat types attaching at undesired translations and reflections. The second technique leverages the power of algorithmic self-assembly to efficiently reuse entire macrotiles which self-assemble in patterns following designed algorithms that dictate the dimensions and patterns of growth.

Using these designs, we demonstrate that in kinetic simulations the systems with reduced numbers of slat types self-assemble more quickly than those with greater numbers. This provides evidence that such optimizations will also result in greater assembly speeds in experimental systems. Furthermore, the reduced numbers of slat types required have the potential to vastly reduce the cost and number of lab steps for crisscross assembly experiments.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases DNA origami, self-assembly, kinetic modeling, computational modeling

Digital Object Identifier 10.4230/LIPIcs.DNA.29.7

Funding *David Doty*: This author's work was supported by NSF grants 2211793, 1900931, and CAREER-1844976.
Hunter Fleming: This author's work was supported in part by NSF grant CAREER-1553166.
Daniel Hader: This author's work was supported in part by NSF grant CAREER-1553166.
Matthew J. Patitz: This author's work was supported in part by NSF grant CAREER-1553166.
Lukas A. Vaughan: This author's work was supported in part by NSF grant CAREER-1553166.

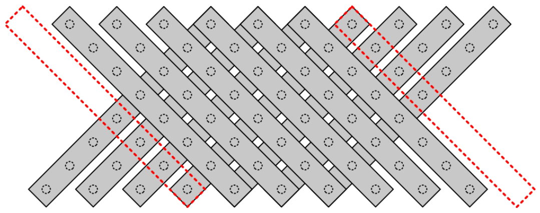


Figure 1 An illustration of a crisscross slat ribbon. Growth of the ribbon occurs in two layers with slats attaching to 4 binding domains presented by previously added slats in the opposite layer.

1 Introduction

In [21, 28], the authors introduced a novel scheme for the seeded self-assembly of DNA-based structures that is extremely resilient to spurious (unseeded) nucleation. In their scheme, structures are formed from individual components called *slats* that can be thought of as long, $1 \times n$ tiles with n binding domains distributed across their lengths. Unlike traditional DNA-based tiles that generally attach in a single plane to at most 2 adjacent tiles as they bind into an assembly, slats attach in multiple layers by the matching of binding domains between the layers. Consequently, whereas traditional tiles typically only coordinate their growth with at most two others, slats can achieve a much higher coordination by spanning across and binding with several slats in the other layer, as illustrated in Figure 1. This increase in coordination (a.k.a., cooperation) makes spontaneous growth away from seeded assemblies much more difficult, since this would require the entropically unfavorable merging of n slats simultaneously, an event with probability exponentially small in n.

While the work of [21, 28] was primarily interested in the robust nucleation properties of this crisscross scheme, here we explore crisscross slats as a means of growing structures that are robust to erroneous attachments while also using greatly reduced numbers of unique types of slats. In [28] they focused on systems of crisscross slats in which each slat is a complete DNA origami structure (depicted in Figure 2), and exhibited growth of structures that were of two types: (1) repeating patterns of slats forming unbounded 1D ribbons and 2D sheets (*periodic* structures), and (2) finite structures composed of unique slat types at each location (*hard-coded*, a.k.a., *fully addressable*, structures). They were able to build hard-coded structures that contained as many as 1,022 unique slat types. Although these structures had few errors (e.g., missing slats or slats in incorrect locations and/or translations or reflections), due to the low concentrations resulting from dilution after mixing so many different structures, the growth was very slow and even needed to be separated into distinct growth stages. In each stage, only a few hundred slat types were added and growth was allowed to continue for multiple days before the slat types for the next stage were added. Additionally, the authors estimated that without automated liquid handlers, manual pipetting of the many strand combinations to create the large numbers of slat types would require about one month of manual effort. Our techniques for lowering the numbers of slat types required to build target structures and patterns are intended to make growth of crisscross slat systems much faster (removing the need for staging) while experiencing similar error rates, and having additional benefits of making system designs cheaper and less labor intensive.

In this paper, we first introduce two models of crisscross slat-based self-assembling systems. These models are based on the abstract Tile Assembly Model (aTAM) and kinetic Tile Assembly Model (kTAM) introduced in [27]. The model based on the aTAM, which we call the *abstract Slat Assembly Model* (aSAM), is a mathematical abstraction suitable for creating and testing high-level designs, especially those of algorithmic self-assembling systems.

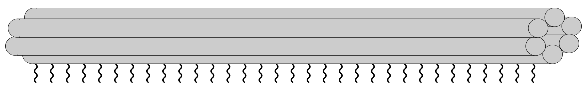


Figure 2 Schematic depiction of DNA origami slat as a 6-helix bundle. Each cylinder represents a DNA double helix, and the wavy lines underneath represent single-stranded "handles" (a.k.a., "glues") that serve as binding domains.

In algorithmic systems, the individual components (in this case, slats), can be thought of as simple program instructions. The binding of each slat effectively uses the domains available for a slat to bind (i.e. those exposed by slats already in the assembly and in the necessary geometric arrangement) to discriminate among slat types, allowing exactly one type to bind. Those initial binding domains can be thought of as the *input*, and the domains of the slat that remain exposed to which later slats can bind as the *outputs*. The design of a slat and its domains dictates the logic that transforms input into output and thus the function of the "instruction". In a computer program, instructions taken from a small number of unique instruction types can be executed many times each, processing information to determine which instructions are next and when the computation should end. In an algorithmic self-assembling system, a small set of slat types do the same thing. Much theoretical work has been done to show the power of algorithmic self-assembly [2,8,11,14,16–18,20,22,27] and that structures can be built with optimally small sets of tile types [3–5,7,23,26] .

The second model we introduce, the *kinetic Slat Assembly Model*, is based on the kTAM and intended to capture more physically realistic aspects of the self-assembly of systems. In this model, slat attachments are modeled as reversible processes, where the forward rates of attachment are dependent upon slat concentrations and the reverse (i.e. detachment) rates are dependent upon the number of bonds formed by a slat. In this way, the model is able to capture a wider spectrum of dynamics and model several types of errors that occur in experimental implementations.

We present two techniques for reducing the number of slat types used by systems. The first technique reduces number of unique slat types used within each (potentially repeating) square region referred to as a *macrotile*. In a slat system where the cooperativity value is n (i.e. each slat needs to bind with n others in order to join an assembly), we call each $n \times n$ region where n slats running horizontally overlap with n slats running vertically a macrotile. We present in Section 4.1 a technique for designing systems making use of multiple slats of the same type in each macrotile, thus reducing the overall slat type count, and then present the results from series of kSAM simulations demonstrating that such systems with fewer slat types can both assemble more quickly and can do so while maintaining low error rates.

Our second method of reducing slat type numbers is algorithmic self-assembly. To demonstrate this, we present our design for a system in which the slats compute the logical `xor` function, resulting in a system producing the discrete self-similar fractal pattern called the Sierpinski triangle [18,24]. We present our macrotile and "tile gadget" design that allows for the type of inter-macrotile cooperativity required for algorithmic growth, then show the results of kSAM simulations of that system implemented with varying levels of cooperativity and intra-macrotile slat counts. The results are very promising and show that growth can be sped up greatly by decreasing slat type counts while also remaining essentially error-free within relatively broad ranges of parameters in comparison with previously designed algorithmic systems.

The organization of this paper is as follows. In Section 2 we define the aSAM and in Section 3 we define the kSAM and discuss various properties of it and how we tuned parameters for our simulations. In Section 4 we present our first method for reducing slat type counts and kSAM simulation results for systems designed using that method. In Section 5 we present our general design for algorithmic systems and a specific design of a system to generate the Sierpinski triangle pattern, then kSAM simulation results of it. Finally, in Section 6 we summarize our results and discuss future directions.

2 The abstract Slat Assembly Model

In this section we briefly introduce the abstract Slat Assembly Model. The abstract Slat Assembly Model (aSAM) is a generalization of the aTAM [27] in which the fundamental components are *slats*, $n \times 1 \times 1$ polyominoes made of cubes in 3D space. Similar to tiles, slats can have *glues* (also referred to as *handles*) on each of their $4n + 2$ faces. Each glue is identified by a *label*, some string of characters (or sometimes a color), and a non-negative integer *strength*. Each glue has a complementary glue which shares its strength. In this paper we will often denote complementary glues using the same labels but with one appended by an asterisk (e.g. "label" and "label*"). Furthermore, we make a distinction between *slats* and *slat types*, the latter being just a description of the glues and length of a slat with no defined position or orientation. The position and orientation of slats is restricted to the 3D integer lattice and two slats which sit incident to one another are said to be *attached* or *bound* with strength s if they share complementary glues of strength s on their abutting faces. An *assembly* is simply a set of non-overlapping slats.

A *slat assembly system* (SAS) consists of a finite set of slat types, an assembly called the *seed assembly* which acts as the starting point for growth, and a positive integer called the *binding threshold*. The binding threshold describes the minimum cumulative glue strength needed for a slat to stably attach to a growing assembly. Growth in the aSAM is described by a sequence of slat attachments. Any slat which could sit on the perimeter of an assembly so that it would be attached to other slats with a cumulative strength meeting the binding threshold is a candidate for attachment, and attachments are assumed to happen non-deterministically. Any assembly which could result from a sequence of slat attachments beginning with the seed assembly of a SAS $\mathcal{S}$ and using only those slat types in the slat set of $\mathcal{S}$ is said to be *producible* in $\mathcal{S}$. Any assembly which permits no additional slat attachments is called *terminal*.

The aSAM is an idealized model intended to abstractly describe the growth of DNA-based slats under ideal conditions, though it makes no attempt to model realist growth dynamics. Rather, the aSAM is useful for designing and understanding complex slat systems on a logical level rather than a physical one. Considering slat systems in the framework of the aSAM allows us to investigate questions such as how many unique handles/glues are necessary to perform a desired task or how many handles could an erroneous slat bind with at any given time. Furthermore, the discrete nature of the aSAM is ideal for computer simulation.

We also consider a restricted version of the aSAM which we call the aSAM$^-$ which limits slat attachments in several ways. These restrictions include slats only being allowed to attach in the planes $z = 0$ and $z = 1$, and the requirement that any two slats sharing the same type will always attach in the same orientation. These restrictions exist to limit our designs to those which grow in a similar manner to those slat system in [28], but also allow for more efficient computer simulation.

2.1 SlatTAS: an aSAM$^-$ simulator

We have developed and freely released the source code for a Python-based graphical simulator for the aSAM$^-$ (and kSAM, see Section 3) called SlatTAS. It can be downloaded from `self-assembly.net` via a link on the page here [15].

2.2 Slat system design parameters

In this section, we provide an example of a SAS and some related terminology that will be used throughout the remaining sections.

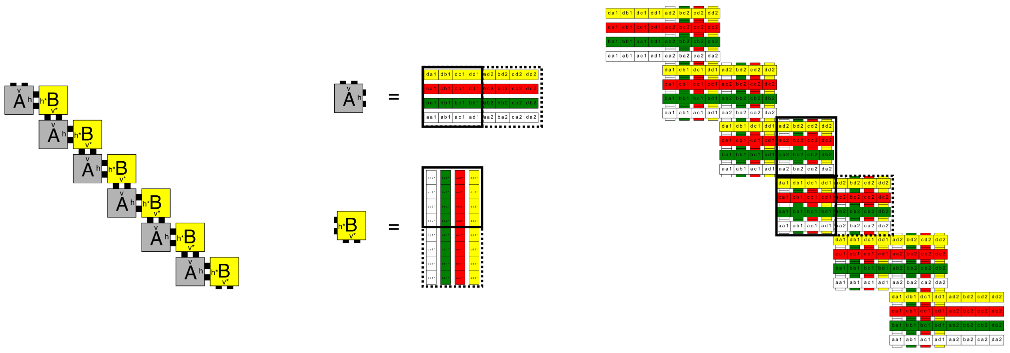


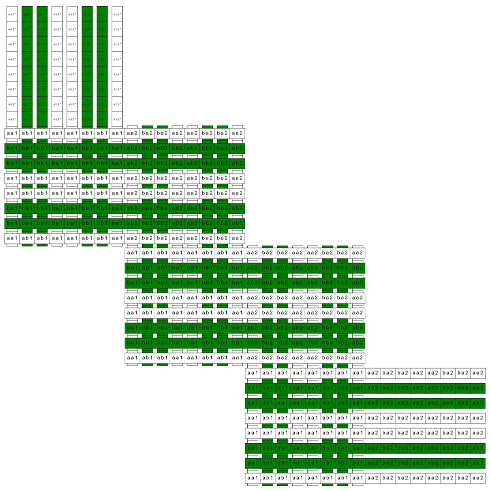
Figure 3 (left) A portion of an example "ribbon" made by a repeating pattern of 2 square tiles, "A" and "B", where one copy of A is used as the seed. (Note that the ribbon would extend infinitely in both directions.) (middle) An example representation of tiles "A" and "B" as slats using macrotiles of size 4. The outlines show the macrotile locations occupied by the slats. (right) A portion of an example system of slats simulating this tile system.

The number of slats to which an incoming slat must bind (since all glues are of strength 1) is equal to the parameter τ and is called the *cooperativity* of a system (a.k.a., *coordination number*, the term used in [21, 28]). Similar to the vast majority of the designs of [28], our systems will be designed so that slats assemble in arrangements which we refer to as forming *macrotiles*. Given a system whose cooperativity is c, a macrotile is simply a $c \times c$ square through which c horizontal slats, and c vertical slats, extend and bind to each other. Multiple macrotiles may be logically combined to form a *tile gadget* (or simply *gadget*) which can be interpreted as a group of slats that "work together" to perform a logical operation, which is often analogous to a tile in an aTAM system that we have converted into a functionally equivalent slat system. For example, the set of four slats in Figure 3 which are shown to "simulate" tile "A" can be thought of as a (relatively) simple gadget whose slats occupy two macrotiles. The macrotile in which the slats initially bind can be thought of as providing the input, and once each slat binds to the four slats in the input macrotile, it extends into the output macrotile. Once all four slats have bound to the input slats, they provide enough domains in each of four horizontal rows for the slats of the gadget simulating the "B" tile attach. This process can alternate between horizontal and vertical gadget attachment infinitely many times, in both the up-left and down-right directions.

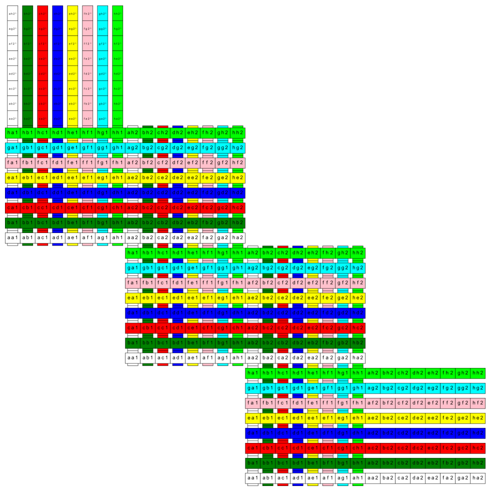
The macrotiles of a gadget can serve as *input*, *output*, and *connector* (to be seen later) macrotiles. We define the *slat count* as the count of unique slat types in each layer of a macrotile. Figures 4a and 4b show example ribbons with cooperativity 8 and slat counts 8 and 2, respectively. We use the term *motif* to refer to the pattern of slat types in a macrotile. For instance, in Figure 4a if the white vertical slats are of type w and the green are of type g, we could denote the motif as $wggwwggw$.

3 The kinetic Slat Assembly Model

To better understand how crisscross slat assemblies grow, we extend the framework of the kinetic Tile-Assembly Model (kTAM) to incorporate slats instead of square tiles. In the kTAM [27], tile attachments are modeled as reversible processes with forward and reverse



(a) Slat count 2 (i.e. each macrotile contains slats of only two different types in each direction).



(b) Slat count 8 (i.e. each slat of each direction in a macrotile is unique).

Figure 4 Example slat ribbons with macrotile size 8 (i.e. cooperativity 8) and varying slat count. Slats running in different directions are of different types, and for each direction each type has a unique color.

rates. Furthermore, the kTAM models *seeded* growth, beginning from a predefined *seed* assembly which is assumed to exist at lower concentrations than the individual tiles. In the kTAM, the forward rate of attachment for tile t has the form $r_f = k_f[t]$, where k_f is the reaction rate constant and $[t]$ is the concentration of t. The kTAM assumes that the rate of tile detachment depends primarily on the number of glue bonds formed by each tile. A tile with only 1 bound glue, for instance, should detach much more quickly than a tile with 2 or more bound glues. The reverse rate therefore depends on the free energy of a typical bond which is denoted ΔG°_{se} where se stands for *sticky end*. Consequently, the reverse rate describing the detachment of a tile with b glue bonds has the form

$$r_{r,b} = k_f u_0 \exp\left(b\frac{\Delta G^\circ_{se}}{RT} + \alpha\right)$$

where u_0 is a standard reference concentration, T is the temperature in Kelvin, R is the molar gas constant, and α is a unitless free energy parameter associated with other factors specific to a particular realization of the tiles. These rates are generally translated into a more symmetric form [10] by defining $\hat{k_f} \overset{\text{def}}{=} u_0 k_f \exp(\alpha)$, $G_{mc} \overset{\text{def}}{=} \alpha - \log(\frac{[t]}{u_0})$, and $G_{se} \overset{\text{def}}{=} -\frac{\Delta G^\circ_{se}}{RT}$. These can then be substituted into the original rate formula to get the following.

$$r_f = \hat{k_f} \exp(-G_{mc}) \qquad\qquad r_{r,b} = \hat{k_f} \exp(-bG_{se})$$

In this form, the parameter G_{mc} describes tile concentrations logarithmically with larger values corresponding to smaller concentrations, and G_{se} describes the free energy of a strength 1 glue with larger values corresponding to stronger bonds. In the kTAM, the ratio G_{mc}/G_{se} plays an important role, analogous to the binding threshold τ in the aTAM. When this ratio is slightly less than 2 for instance, growth in the kTAM proceeds much like it would in a binding threshold 2 aTAM system with the same tiles. This is because at this ratio, tiles bound with strength 2 dissociate at a rate just barely less than tiles attach to the assembly while tiles bound with strength 1 or 0 dissociate with a significantly higher rate.

Consequently, a correctly bound tile (matching 2 glues) is likely to remain attached to the assembly for long enough for additional tile attachments to occur, matching glues with the correct tile and subsequently increasing its number of bonds so that it is held stably to the attachment. Incorrect tiles (matching 1 glue or fewer) on the other hand detach much more rapidly and very likely before any additional tile attachments occur. When this ratio is exactly equal to 2, or more generally the cooperativity of the tile system, then the kTAM describes how the system behaves at the melting temperature, i.e. even the strongest bound tiles detach just as often as tiles attach to the assembly. In general, the ratio of G_{mc} to G_{se} is often more pertinent when describing the expected dynamics of a system than the values of the individual parameters, though the parameters themselves are necessary for computer simulation of the kTAM. For a more detailed description of the kTAM and subsequent analyses for square tiles, see [10, 27].

Despite making several simplifying assumptions, the kTAM has been broadly successful in realistically describing growth and error phenomena in a variety of tile-based schemes of DNA self-asssembly [6, 9]. The model captures many critical features of physical tiles while remaining simple enough for meaningful mathematical manipulation and efficient computer simulation using the Gillespie algorithm [12]. Generalizing the kTAM to incorporate slats rather than square tiles is, in principle, as straightforward as it sounds. Instead of square tiles which occupy only a single grid location and contain at most 4 glues, we consider slats which can occupy several adjacent grid locations and have a number of glues proportional to their length. For purposes of brevity and clarity, we will refer to this generalization as the *kinetic Slat Assembly Model* or kSAM to distinguish it from the kTAM for square tiles.

3.1 Finding appropriate parameter values for physical slats

While many important properties of a kSAM system can be inferred from the ratio G_{mc}/G_{se}, it's useful to find specific values for these individual parameters which yield simulations with accurate growth rates. Other than α which describes an entropic cost associated with a specific implementation of slats, all of the factors defining the parameters G_{mc}, G_{se}, and $\hat{k_f}$ such as temperature and slat concentrations are generally known. We can roughly estimate the value of α by noting that, for slat systems with cooperativity c, the melting temperature should correspond to values for G_{mc} and G_{se} such that $G_{mc} = cG_{se}$. Consequently, by substituting G_{mc} and G_{se} with their definitions we find that

$$\alpha - \log\left(\frac{[s]}{u_0}\right) = -c\frac{\Delta G^\circ_{se}}{RT}.$$

Furthermore, $\Delta G^\circ_{se} = \Delta H^\circ_{se} - T\Delta S^\circ_{se}$ where ΔH°_{se} and ΔS°_{se} are the enthalpy and entropy associated with a single sticky end bond whose values can be approximated using the nearest neighbor model. Rearranging the above equation, thus yields the following expression for α which assumes that T is the melting temperature of the system.

$$\alpha = \frac{c}{R}\left(\Delta S^\circ_{se} - \frac{\Delta H^\circ_{se}}{T}\right) + \log\left(\frac{[s]}{u_0}\right)$$

In [28], the authors determined the melting temperature for origami slat ribbons using cooperativity values of 8 and 16 and using handle lengths of 6, 7, and 8 nucleotides. Using their values for melting temperature and the corresponding slat concentrations, and applying the nearest neighbor model to their handle sequences therefore allows us to find a value for α. Using the handle sequences from [28] we calculated typical values for 7nt handles to be about $\Delta H^\circ_{se} \approx -47\text{kcal}\,\text{mol}^{-1}$ and $\Delta S^\circ_{se} \approx -142\text{cal}\,\text{mol}^{-1}\,\text{K}^{-1}$. For their slat concentration

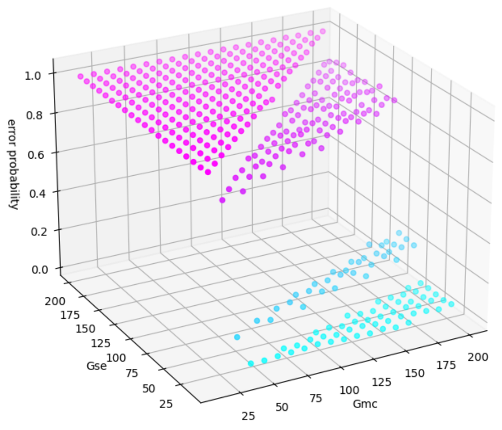


Figure 5 Error rates for kinetic simulations of cooperativity 16 ribbons at G_{mc} and G_{se} values ranging from 10 to 200. The general shape of the error curve matches the trends predicted by the kinetic trapping model with sharp jumps as the ratio G_{mc}/G_{se} crosses an integer value. Furthermore, when $G_{mc} < G_{se}$ the error probability is 1.

of 20nM and melting temperature of $42.1°\text{C} = 315.25\text{K}$, these values yield a value for α of around 39. Admittedly, this approximation isn't perfect; for one, the formula for α is quite sensitive to the values of ΔH°_{se} and ΔS°_{se} which can vary a bit for the same glue sequence depending on the specific duplex table used in the nearest neighbor model calculations. For our simulations we chose to use $\alpha = 40$, though we note that for our purposes the specific value for α makes little difference in the error dynamics of our simulated slat systems.

3.2 A kSAM simulator

We have developed and freely released the source code for a C++ based stochastic simulator for both the aSAM and kSAM, which has been highly optimized for faster simulation than the existing SlatTAS [15]. It can be downloaded from `self-assembly.net` via a link on the page here [13]. More generally, our simulator is capable of handling arbitrary 3D polyomino-shaped tiles (including square/cube tiles) and can be configured to simulate 2D and almost-3D (with only 2 layers in the z-axis) systems as well by use of a dimension restriction which allows for specifying minimum and maximum allowed coordinates in all 3 dimensions if desired. The simulator can also use multiple threads to simulate a system ensemble. All kinetic simulations in this paper were performed using our simulator in kinetic mode and our error metrics were calculated using reference assemblies generated by our simulator in abstract mode.

4 Optimized implementation of crisscross ribbons via slat type reuse

In this section, we describe our first technique for reducing slat type counts, which involves slat type reuse within macrotiles. We then describe the range of systems with varying amounts of such intra-macrotile slat reuse that we designed and then tested via kSAM simulations, and discuss the results of those simulations.

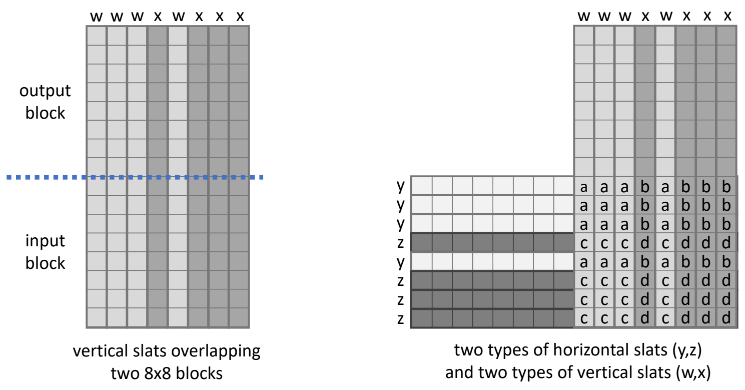


Figure 6 Example using two types of slats in each direction for 8×8 macrotiles. (*left*): Two types of vertical slats, w and x, with four copies of each used for the eight total vertical slats used in the input and output macrotiles. (*right*): Two types of horizontal slats, y and z, where input glues of w and x bind to output glues of y and z. This illustrates how slat re-use within a macrotile forces re-use of glues. Where slats w and y overlap, they must have a complementary glue (a on w and a^* on y, latter not depicted since the output glues of the y slat are underneath the w and x slats). By symmetry, that same glue a is forced to be used wherever slats w and y intersect. In this situation, the maximum number of glues that can possibly be used is only four: a, b, c, d, increasing the potential for misaligned slats to have complementary glues in adjacent locations over systems which use larger numbers of glues.

4.1 Slat layout in macrotile designs

Most assembly schemes in [28] consider slat assembly in a modular way, by partitioning the integer lattice $\mathbb{Z}^2$ into $n \times n$ macrotiles. The n slats that assemble to fill in an $n \times n$ macrotile can be considered as "simulating" a square tile in the abstract and kinetic Tile Assembly Models [27]. In [28], a unique slat type was used in every one of the n relative positions within a macrotile (even for periodic structures such as infinite 1D ribbons and 2D sheets that used the rectangular macrotile growth pattern and repeated the same set of slats in different macrotiles). However, we have analyzed patterns of possible slat reuse within macrotiles and developed a technique for greatly reducing how many are required in each while seeking to maintain (mostly) error-free growth, which we now describe.

Figure 6 shows an example of using only two types of slats in each direction when we use cooperativity 8 to have 8×8 macrotiles. Its caption explains how this slat reuse forces the reuse of glues. This reuse of glues in turn means we must confront the possibility that slats can bind "strongly" somewhere that they should not (perhaps flipped and/or translated such that they are not in alignment with macrotile boundaries) due to complementarity of a *partial* subset of glues. Figure 7 shows some potential scenarios with erroneous binding. Although we declare in the abstract aSAM model that no binding occurs if the number of matching glues is less than the cooperativity value n, in the kSAM and in actual experiments it is possible for such situations to allow for slat bindings with "close" to n bonds (although with expected duration of attachment to diminish as the distance from n increases). Therefore, we strive to ensure that slat and glue patterns are designed so that the maximum strength of any such incorrect slat binding is minimized. We analyze motif patterns to detect and minimize a property we call *auto-correlation* in order to minimize the errors that occur during self-assembly.

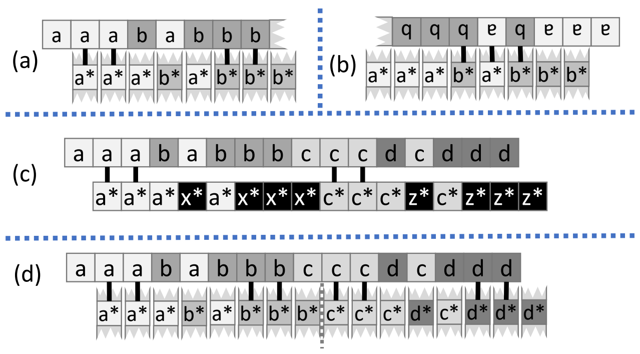

(a)
a a a b a b b b
a* a* a* b* a* b* b* b*
(b)
a* a* a* b* a* b* b* b*
(c)
a a a b a b b b c c c d c d d d
a* a* a* x* a* x* x* x* c* c* c* z* c* z* z* z*
(d)
a a a b a b b b c c c d c d d d
a* a* a* b* a* b* b* b* c* c* c* d* c* d* d* d*

Figure 7 Design of slat arrangement to minimize overlap of complementary glues in unintended slat bindings. (a): The input glues of a horizontal slat (*aaababbb*; output glues on right not depicted) binding with incorrect translation to the output glues of eight vertical slats presenting glues $a^*a^*a^*b^*a^*b^*b^*b^*$. This arrangement of input glues has *auto-correlation* 4: this translation makes 4 of the glues bind, and every other nonzero translation also has at most 4 matching glues. This means the slat could erroneously bind with strength 4 in the incorrect x position. (b): A string of input glues for a slat with reversed auto-correlation 3. Such a slat could bind with incorrect 180 degree rotation with strength 3. (c): Two full slats binding erroneously to each other. A horizontal slat (on top; input glues a, b on left, output glues c, d on right) and a vertical slat (on bottom; output glues a^*, x^* on left, input glues c^*, z^* on right) bind. The potential for such binding occurs in situations such as the repeating ribbon of Figure 4a, where (in the normal expected binding) a horizontal slat type binds to a vertical slat type, then the same vertical slat type binds to the horizontal slat. By our glue design; in this scenario at most one glue can be shared on each side, e.g., since they share a on the left, they have different glues (b vs. x) for the other type on the left. This check was done in [28]. (d): Similar to part (a), but considers output glues of a slat also. In this example, the left-side block are input glues for the horizontal slat, and the right-side block are the outputs. This describes the scenario where, after vertical slats begin binding on the right (with less than total cooperativity) to horizontal slats (not shown), before the block is complete with horizontal slats, a correct horizontal slat with incorrect x-translation "backfills" in the block.

4.2 Testing slat reuse via kinetic simulations

In [28], the authors, among several other experiments, grew a variety of ribbons using DNA-origami crisscross slats and investigated how their growth was affected by the number of unique slat types used. Specifically, these ribbons, all of which used a cooperativity value of 16, were designed to use either 8, 16, 32, or 64 unique slat types in each of the two layers of growth, attaching periodically. They evaluated the growth of these ribbons both in the situation where the concentration of individual slats was kept constant, and where the total concentration of all slats was held constant (i.e. smaller slat counts having higher per-slat concentrations). While the growth of these ribbons was generally similar when individual slat concentrations were maintained, they did observe a significant decrease in growth as slat count increased when total slat concentrations were fixed.

Here we explore the effects of slat count on ribbon growth conversely, to determine the extent to which we can expect to decrease slat counts while preserving rapid correct growth of the ribbons. To do this, we designed several cooperativity 16 ribbon systems with decreasing slat count and evaluated their growth dynamics within the framework of the kSAM. Our systems included ribbons with 2, 4, 8, and 16 unique slat types for each layer and we performed kSAM simulations of these systems both with individual slat concentrations held constant and with total slat concentrations held constant. The ribbon systems we used are illustrated in Figure 8 and additional info can be seen in Table 1. We chose to implement zig-zag ribbons which more closely match our motif for algorithmic growth than the "staggered" ribbons of some of the designs in [28].

We simulated each of these systems at a fixed value of G_{mc} and varied G_{se}. In principle, this corresponds to growing the slat systems with fixed slat concentrations at a variety of temperatures. The specific value of G_{mc} used was 58 which corresponds to a slat concentration

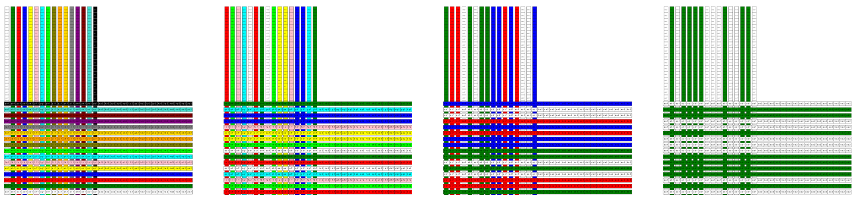


Figure 8 The cooperativity 16 ribbons simulated in the kSAM with various slat counts. Slat counts from left to right: 16, 8, 4, 2.

Table 1 For the cooperativity 16 ribbon systems tested via kSAM simulations with given slat counts, the motifs, auto-correlations and cross-correlations.

Slat count	Motif	Auto-correlation	Cross-correlation
16	`abcdefghijklmnop`	1	2
8	`chfgacbaheefddgb`	2	4
4	`bccababbddcdcaad`	4	4
2	`ababbbbaaabaabba`	8	8

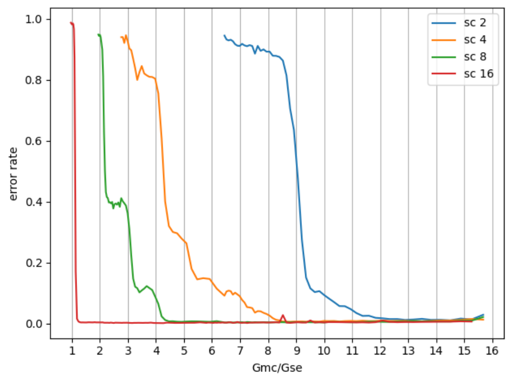
of 15nM using the value for α derived in Section 3.1. Values of G_{se} were chosen so that G_{mc}/G_{se} values spanned the range from 1 to 16. For each ribbon, we wanted to simulate 2 situations: (1) with individual slat counts kept constant at 15nM for all ribbons, and (2) with total slat concentrations held constant at 480nM across all ribbons. To keep total slat concentrations fixed, systems with smaller slat counts had their slat concentrations adjusted[1]. Note that for both the fixed individual slat concentrations and fixed total concentrations, concentrations are identical for the slat count 16 ribbons, so those were only simulated once.

To estimate the rate of erroneous slat attachment, we additionally simulated each system in the error-free aSAM with a binding threshold of 16. This resulted in a reference assembly containing only correctly attached slats to which our kinetic simulations could be compared. We then consider a slat to be correct in the kinetic simulation when a corresponding slat, of the same type in the same translation, exists in the reference assembly. To estimate growth rates in our simulated systems, we note that the time between events during a kinetic simulation can be determined by sampling a value of Δt from an exponential distribution whose rate parameter is the net rate of any event (attachment or detachment) occurring [27]. The growth time of a simulation can thus be calculated by summing the sampled Δt values for each event which occurs in the simulation. Growth rate is then simply estimated as the number of slats which attached by the total growth time.

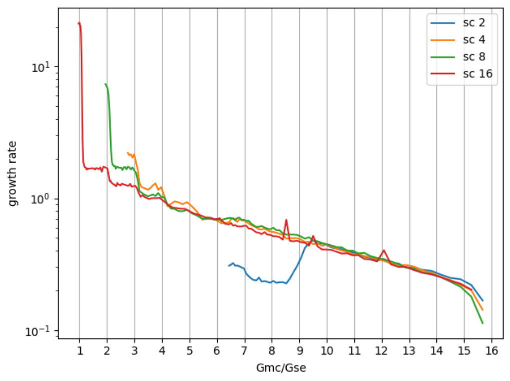
4.3 Results for ribbons simulated with fixed individual slat concentrations

Figures 9a and 9b describe the estimated error and growth rates of the various ribbons with a fixed individual slat concentration. Note that $G_{mc}/G_{se} = 16$ corresponds to the melting temperature since with those parameters, the rate of slat attachment is equal to the rate of detachment of slats with 16 bound glues. Consequently, the growth rate beyond that point will be 0. Near the melting temperature, all ribbons exhibit very few erroneous

[1] In our simulation code, this is implemented as a multiplier to the forward rate, rather than adjusting G_{mc}, though the effect is the same.



(a) Simulated error rates of our cooperativity 16 ribbons when individual slat concentrations are held constant.



(b) Simulated growth rates of our cooperativity 16 ribbons on a log scale when individual slat concentrations are held constant.

Figure 9 Results for kinetic simulations of cooperativity 16 ribbons with fixed individual slat concentrations. Results represent averages over 100 simulations.

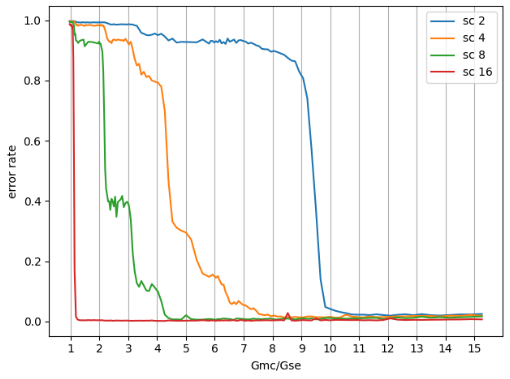
slat attachments. Interestingly, the error rate for all ribbons remain essentially at 0 for $G_{mc}/G_{se} > 12$. Additionally, growth rates for all ribbons in this range are essentially equivalent between the differening slat counts. Together these results suggest that there exists a range of temperatures in which the slat count 2 ribbons grow just as well as the slat count 4 and 8 ribbons. As discussed further in Section 7.1 of the appendix, the growth rate of the slat count 2 ribbons interestingly decreases when G_{mc}/G_{se} drows below around 9. This seems to be due to a high frequency of erroneous slat attachments which quickly occupy most available stable attachment sites preventing further growth.

In general, the error rates for the systems drastically increase as the G_{mc}/G_{se} ratio drops to the value corresponding to the auto-correlation values of the systems. This is because for each system there exist one or more slats that can bind while in an incorrect/misaligned location with a number of glues equal to that system's auto-correlation value, and when that is near or equal G_{mc}/G_{se}, those attachments are relatively stable. Thus, the prevalence of errors rapidly increases near those values. Due to space constraints, more details of the types of errors observed can be found in Section 7.1 of the appendix.

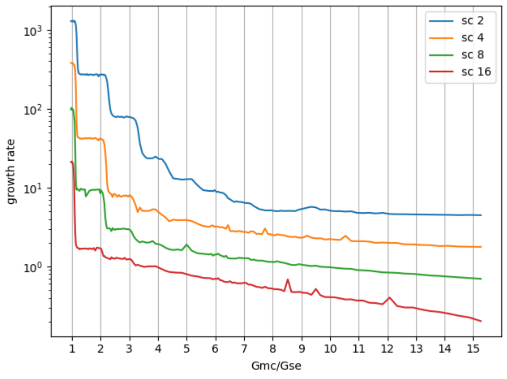
The results of these kSAM simulations imply that there are ranges of G_{mc} and G_{se} where ribbon growth can remain almost entirely error-free even when the slat counts are reduced to the nearly optimal value of 2. (Note that any system using cooperativity n with a slat count of 1 would necessarily have auto-correlation value of $n-1$, and thus a very narrow possible range for G_{mc}/G_{se} with low errors.) Additionally, even with individual slat concentrations held constant across systems, the growth rates were roughly equivalent across wide ranges of G_{mc}/G_{se}, meaning that the lower absolute concentrations of slats (and thus lower arrival rates of slats at frontier locations) was balanced by the higher likelihood of a slat type being the correct type for a location in which it randomly arrives.

4.4 Results for ribbons simulated with fixed total slat concentrations

In addition to simulating the ribbons with fixed individual slat concentrations, we also simulated the ribbons with total concentrations fixed. That is, systems using fewer unique slats had higher concentrations for each slat. For these experiments, we used the same slat designs as in the previous section (with details shown in Table 1), changing only slat



(a) Error rates for cooperativity 16 ribbons using slat counts of 2, 4, 8, and 16 using a fixed total concentration of all slats in each system. Notice that despite having different concentrations, the results are near identical to those in Figure 9a.



(b) Growth rates for cooperativity 16 ribbons using slat counts of 2, 4, 8, and 16 with fixed total concentration of all tiles in each system. Results are plotted on a log scale. Growth rates are larger for smaller slat counts suggesting that decreasing slat counts allows for faster growth of ribbons.

Figure 10 Results for kinetic simulations of cooperativity 16 ribbons with fixed total slat concentrations. Results are averaged over 100 simulations.

concentrations. This is in line with the ribbon experiments performed in [28]. Error rates and growth rates are calculated using the same method for the analogous results with fixed individual slat counts. Our results are summarized in Figure 10a and Figure 10b. While decreasing slat counts caused ribbons to have narrower regions of correct growth, our results show that under the right conditions, all ribbons were able to grow essentially error free. Moreover, in our simulations, halving slat counts lead to growth rates increasing by an average factor of 2.35 at $G_{mc}/G_{se} = 11$. In fact over the range of G_{mc}/G_{se} values where all ribbons grew with little error, (i.e. above 11), the ratio of growth rates between ribbons whose slat counts differed by a factor of 2 was consistently greater than 2 as illustrated in Figure 19. In other words, halving slat counts more than doubled growth rates in our simulations. Whether these results would translate into the lab is unclear, but regardless our results suggest that decreasing slat counts can be a powerful tool for improving growth rates without sacrificing much error. Importantly, this technique for reducing slat type counts is applicable in general, to all designs using macrotiles. (For more detailed examination of some of the errors observed, please see Section 7.2.)

5 Algorithmic self-assembly using crisscross slats

An $n \times n$ square composed of square tiles in the aTAM can self-assemble in a system which uses a unique tile type at each location (a so-called *fully-addressable* or *hard-coded* system), requiring n^2 unique tile types. However, an algorithmic system can form a structure of the exact same shape using an optimal (via an information theoretic argument) $\frac{\log(n)}{\log(\log(n))}$ tile types [1, 25], meaning that the hard-coded system uses exponentially more tile types than the algorithmic system. As previously mentioned, reducing the number of unique component types has the benefits of making a physical implementation via DNA (1) cheaper, (2) faster, and (3) require fewer unique domains thus making their individual binding characteristics more uniform. However, in order to leverage the power of algorithmic self-assembly, a slightly different notion of cooperativity than previously discussed is necessary. While the previously

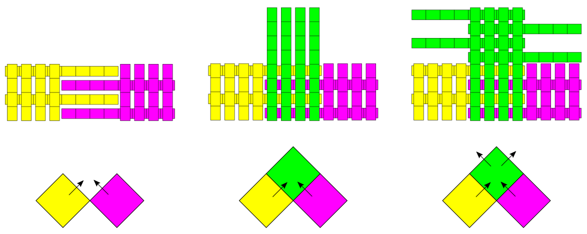


Figure 11 An example of "algorithmic" cooperation, i.e. the inputs of one gadget's slats are the outputs of slats from two different gadgets. In this case, the yellow slats belong to one gadget and the pink to another. They overlap in the location of the central macrotile to provide input to the green slats. This is often referred to as *across-the-gap* cooperation. This process is analogous to the algorithmic cooperation realized by square yellow and pink tiles cooperatively binding to a square green tile.

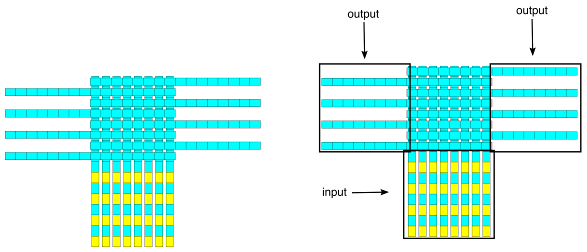


Figure 12 An example gadget that receives its input in a macrotile at the bottom, then has slats which propagate output to macrotiles to the top left and right.

discussed notion of cooperativity concerned the binding of a single slat to multiple other (orthogonally oriented) slats in order to attach to an assembly, in the examples shown (e.g. Figure 3) all slats bound to as input for one gadget were acting as output for a single other gadget. This effectively provides only a single logical input to direct the growth of each gadget. Algorithmic growth, on the other hand, requires that some gadgets receive input from at least two distinct gadgets, allowing the combined information from both input gadgets to direct the growth of the new gadget. (This was shown to be necessary by the requirement of a minimum temperature value of 2 in the aTAM [19, 20].) An example of such algorithmic cooperativity implemented via slats in 4×4 macrotiles can be seen in Figure 11.

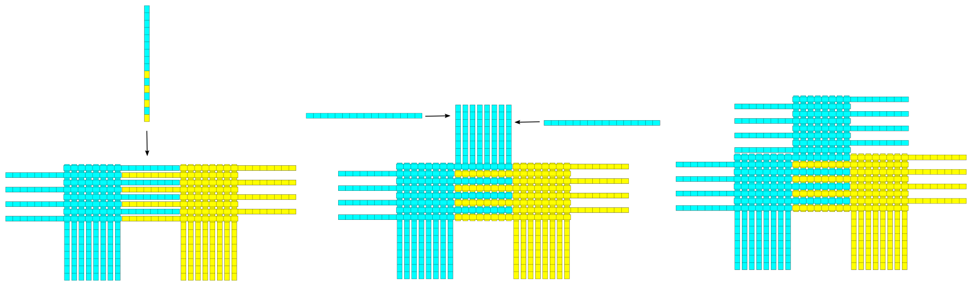


Figure 13 Order of growth of a gadget using algorithmic cooperation.

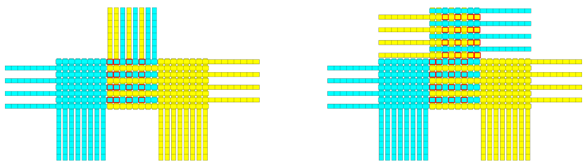
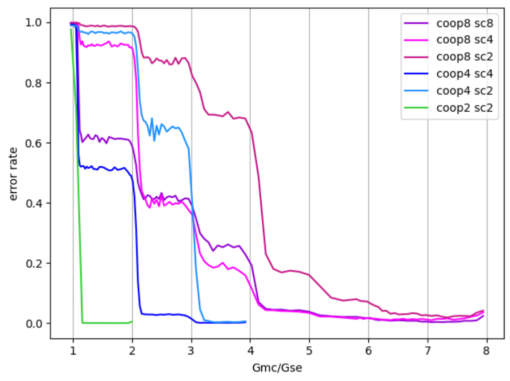


Figure 14 Example of algorithmic error propagation through a gadget.

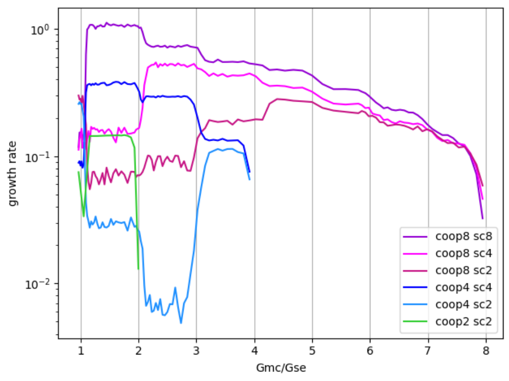
5.1 Kinetic simulations and error analyses of algorithmic systems

We hypothesized that our design for algorithmic systems using slats arranged in gadgets composed of macrotiles should exhibit strong proofreading characteristics. The reasoning for this is depicted in Figures 12–14 where (1) the input and output regions of a gadget are depicted (Figure 12), (2) the ordering of correct growth is shown (Figure 13), and (3) the number of concurrent errors that would need to occur and persist in order for slats to bind with even half of the designed binding strength in order to propagate algorithmic errors is exemplified Figure 14. In general, for a macrotile scheme at cooperativity k (i.e. using slats of length $2k$), at least k erroneous half-strength slat attachments are required for at least half of a macrotile's outputs to encode the wrong value. While, this is the same number of erroneous attachments necessary to invalidate a square tile system using $k \times k$ block-replacement, our macrotiles consist of only k slats rather than k^2 square tiles. In other words, a square tile block-replacement scheme requires a quadratic increase in tile complexity, compared to a linear increase for our slat based macrotiles, to achieve the same amount of proofreading.

The discrete self-similar fractal pattern known as the Sierpinski triangle is a well-studied pattern in aTAM and kTAM self-assembly, both theoretically [18] and experimentally [24]. This is due to its relatively simple algorithmic logic (each location represents a 0 or 1 bit value that is the `xor` of the bits of its two input neighbors) that results in an infinite, aperiodic pattern known as a discrete self-similar fractal. (See Section 7.3 for depictions of the Sierpinski triangle.) We designed several macrotile-based algorithmic slat systems which grow to form a Sierpinski triangle pattern using cooperativity values of 2, 4, and 8. In total we simulated 6 designs, permuting cooperativity and slat counts. For cooperativity 8 triangles we tested slat counts of 2, 4, and 8; for cooperativity 4 triangles we tested slat counts of 2 and 4, and our cooperativity 2 triangles used a slat count of 2. Concentrations of individual slats was kept constant at 15nM per slat. The results of the simulations are summerized in Figure 15a. Our results show that while decreasing slat counts increased error rates consistently, there were still ranges of conditions under which all triangles grew with little error. Interestingly, for the cooperativity 4 triangles, using a slat count of 2, growth rates slowed significantly below a G_{mc}/G_{se} of 3. Much like the slat count 2 ribbons in Section 4, we suspect that this is due to erroneous slats quickly filling up most of the stable attachment sites preventing further growth. This is supported by the fact that growth rates increase below a G_{mc}/G_{se} of 2 where stable attachment sites require fewer correct glues. Regardless, our triangle systems exhibit a remarkable tolerance to infrequent errors as can be seen in Figure 23 and Figure 24. These results support the idea that even for algorithmic systems, under ideal conditions, reducing slat counts is possible without sacrificing much in terms of error.



(a) Rate of erroneous slat attachment in kinetic simulations of our algorithmic slat systems as a function of G_{mc}/G_{se}. All systems exibited little error in conditions near their respective melting points, where G_{mc}/G_{se} equals the cooperativity value of the system.



(b) Log growth rate during kinetic simulations of our algorithmic slat systems as a function of G_{mc}/G_{se}. Here growth rate is calculated as the number of slat additions per unit of simulated time.

Figure 15 Results for kinetic simulations of Sierpinski triangle systems. Each data point represents the average of 100 simulations.

6 Conclusions and Future Work

In this paper, we have introduced an abstract mathematical model for self-assembling systems composed of slats that can combine in layers using high levels of cooperativity, and shown how to design algorithmic self-assembling systems of slats within it. We have also introduced a more physically realistic “kinetic” model capable of capturing many types of errors that occur in laboratory implementations of self-assembling systems, and showed that we were able to tune parameters of simulations within that model to match results of systems from [28]. We then presented a technique for reducing the number of unique slat types required to build ribbon structures (that generalizes to all macrotile-based designs) and showed via simulations that growth rates can be greatly accelerated while low error rates are maintained. Finally, we presented a technique for designing algorithmic self-assembling systems of slats and demonstrated it by designing a system that generates the Sierpinski triangle pattern, which we also simulated to show that extremely low error rates can be maintained. Due to the fact that algorithmic systems are capable of self-assembling structures while utilizing only a logarithmic number of unique components relative to hard-coded structures, and combined with the first technique for reducing slat types, our designs result in slat systems with dramatically fewer slat types than previous designs. Simulations show that these systems will therefore self-assemble much faster, and they will also be much easier and cheaper to implement.

We will be implementing these and similar systems using DNA origami slats and comparing the laboratory results to the results of our simulations, then updating the model (and our designs) as necessary to refine the designs. Future work could include new designs of macrotiles so that the inter-macrotile cooperativity needed for algorithmic self-assembly is achieved through different slat patterns and/or orientations, and then simulations and laboratory experiments to see which are best. Also, further examination of the types of errors that occur during simulations and laboratory experiments may result in additional checks and optimizations to be made for designs. It is our hope that crisscross slat based self-assembling systems will achieve high-levels of algorithmic sophistication while maintaining low enough levels of algorithmic errors to realize much more of the theoretical potential that has been pursued for so long.

References

1 Leonard Adleman, Qi Cheng, Ashish Goel, and Ming-Deh Huang. Running time and program size for self-assembled squares. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 740–748, Hersonissos, Greece, 2001. `doi:10.1145/380752.380881`.

2 Nathaniel Bryans, Ehsan Chiniforooshan, David Doty, Lila Kari, and Shinnosuke Seki. The power of nondeterminism in self-assembly. *Theory of Computing*, 9:1–29, 2013. `doi:10.4086/toc.2013.v009a001`.

3 Sarah Cannon, Erik D. Demaine, Martin L. Demaine, Sarah Eisenstat, Matthew J. Patitz, Robert T. Schweller, Scott M. Summers, and Andrew Winslow. Two hands are better than one (up to constant factors): Self-assembly in the 2HAM vs. aTAM. In Natacha Portier and Thomas Wilke, editors, *STACS*, volume 20 of *LIPIcs*, pages 172–184, 2013.

4 Cameron T. Chalk, Eric Martinez, Robert T. Schweller, Luis Vega, Andrew Winslow, and Tim Wylie. Optimal staged self-assembly of general shapes. *Algorithmica*, 80(4):1383–1409, 2018. `doi:10.1007/s00453-017-0318-0`.

5 Ho-Lin Chen, David Doty, and Shinnosuke Seki. Program size and temperature in self-assembly. In *ISAAC 2011: Proceedings of the 22nd International Symposium on Algorithms and Computation*, volume 7074 of *Lecture Notes in Computer Science*, pages 445–453. Springer-Verlag, 2011.

6 Ho-Lin Chen, Rebecca Schulman, Ashish Goel, and Erik Winfree. Reducing facet nucleation during algorithmic self-assembly. *Nano Letters*, 7(9):2913–2919, 2007.

7 E. D. Demaine, M. L. Demaine, S. P. Fekete, M. J. Patitz, R. T. Schweller, A. Winslow, and D. Woods. One tile to rule them all: Simulating any tile assembly system with a single universal tile. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP 2014)*, IT University of Copenhagen, Denmark, July 8-11, 2014, volume 8572 of *LNCS*, pages 368–379, 2014.

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. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science*, FOCS 2012, pages 302–310, 2012.

9 Constantine G Evans, Rizal F Hariadi, and Erik Winfree. Direct atomic force microscopy observation of dna tile crystal growth at the single-molecule level. *Journal of the American Chemical Society*, 134(25):10485–10492, 2012.

10 Constantine G Evans and Erik Winfree. Physical principles for DNA tile self-assembly. *Chemical Society Reviews*, 46(12):3808–3829, 2017.

11 David Furcy, Scott M Summers, and Christian Wendlandt. Self-assembly of and optimal encoding within thin rectangles at temperature-1 in 3d. *Theoretical Computer Science*, 872:55–78, 2021.

12 Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The journal of physical chemistry*, 81(25):2340–2361, 1977.

13 Daniel Hader. RodSim: An Optimized Simulator for the kSAM. `http://self-assembly.net/wiki/index.php?title=RodSim`, 2023. [Online; accessed 28-April-2023].

14 Daniel Hader, Aaron Koch, Matthew J. Patitz, and Michael Sharp. The impacts of dimensionality, diffusion, and directedness on intrinsic universality in the abstract tile assembly model. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2607–2624. SIAM, 2020.

15 Daniel Hader and Matthew J. Patitz. SlatTAS: A Graphical Simulator for the aSAM$^-$. `http://self-assembly.net/wiki/index.php?title=SlatTAS`, 2023. [Online; accessed 28-April-2023].
16 Daniel Hader, Matthew J Patitz, and Scott M Summers. Fractal dimension of assemblies in the abstract tile assembly model. In *International Conference on Unconventional Computation and Natural Computation*, pages 116–130. Springer, 2021.
17 James I. Lathrop, Jack H. Lutz, Matthew J. Patitz, and Scott M. Summers. Computability and complexity in self-assembly. *Theory Comput. Syst.*, 48(3):617–647, 2011. `doi:10.1007/s00224-010-9252-0`.
18 James I. Lathrop, Jack H. Lutz, and Scott M. Summers. Strict self-assembly of discrete Sierpinski triangles. *Theoretical Computer Science*, 410:384–405, 2009.
19 Pierre-Étienne Meunier, Damien Regnault, and Damien Woods. The program-size complexity of self-assembled paths. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 727–737, 2020. `doi:10.1145/3357713.3384263`.
20 Pierre-Étienne Meunier and Damien Woods. The non-cooperative tile assembly model is not intrinsically universal or capable of bounded Turing machine simulation. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 328–341, 2017. `doi:10.1145/3055399.3055446`.
21 Dionis Minev, Christopher M Wintersinger, Anastasia Ershova, and William M Shih. Robust nucleation control via crisscross polymerization of highly coordinated DNA slats. *Nature Communications*, 12(1):1–9, 2021.
22 Matthew J. Patitz and Scott M. Summers. Self-assembly of decidable sets. *Natural Computing*, 10(2):853–877, 2011. `doi:10.1007/s11047-010-9218-9`.
23 Paul W. K. Rothemund. *Theory and Experiments in Algorithmic Self-Assembly*. PhD thesis, University of Southern California, December 2001.
24 Paul W. K Rothemund, Nick Papadakis, and Erik Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biol*, 2(12):e424, December 2004.
25 Paul W. K. Rothemund and Erik Winfree. The program-size complexity of self-assembled squares (extended abstract). In *STOC '00: Proceedings of the thirty-second annual ACM Symposium on Theory of Computing*, pages 459–468, Portland, Oregon, United States, 2000. ACM.
26 David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. *SIAM Journal on Computing*, 36(6):1544–1569, 2007. `doi:10.1137/S0097539704446712`.
27 Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, June 1998.
28 Christopher M Wintersinger, Dionis Minev, Anastasia Ershova, Hiroshi M Sasaki, Gokul Gowri, Jonathan F Berengut, F Eduardo Corea-Dilbert, Peng Yin, and William M Shih. Multi-micron crisscross structures grown from dna-origami slats. *Nature Nanotechnology*, pages 1–9, 2022.

7 Appendix

This section contains technical details of results omitted from the main body of the paper due to space constraints.

7.1 Details regarding ribbon systems with fixed individual slat concentrations

Here we provide more in-depth analysis of the types and causes of errors that we observed in the simulations of ribbons with varying amount of slat reuse in Section 4.3. We attempt to give our intuitions for why we think error and growth rates trend as they do. For example, the auto-correlation value for the slat count 2 system is 8, and for that system error rates begin to increase slowly as G_{mc}/G_{se} drops below about 12 until around 9.5 where it suddenly begins to spike. However, and importantly for continued growth of the ribbons, while the error rate in this region is distinctly larger than 0, we note that the errors present are generally isolated and do little to affect the overall growth of the ribbon. Figure 16 depicts a section taken from a typical slat count 2 ribbon at $G_{mc}/G_{se} = 10$. Erroneous slats can be seen clearly in the assembly as they are generally not aligned with the others, but because only a small number are present in each block, the further attachment of correct slats is not impeded significantly.

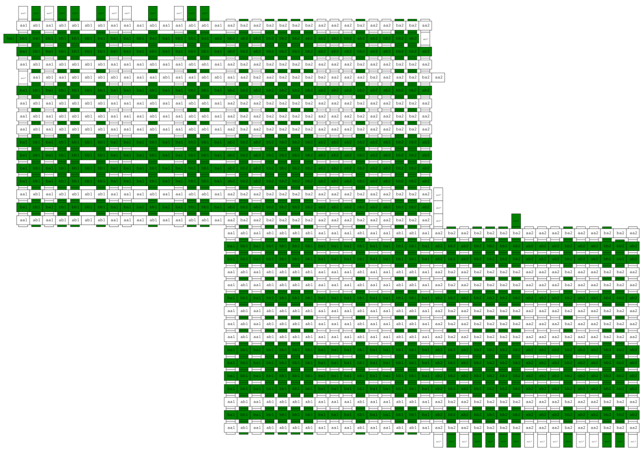
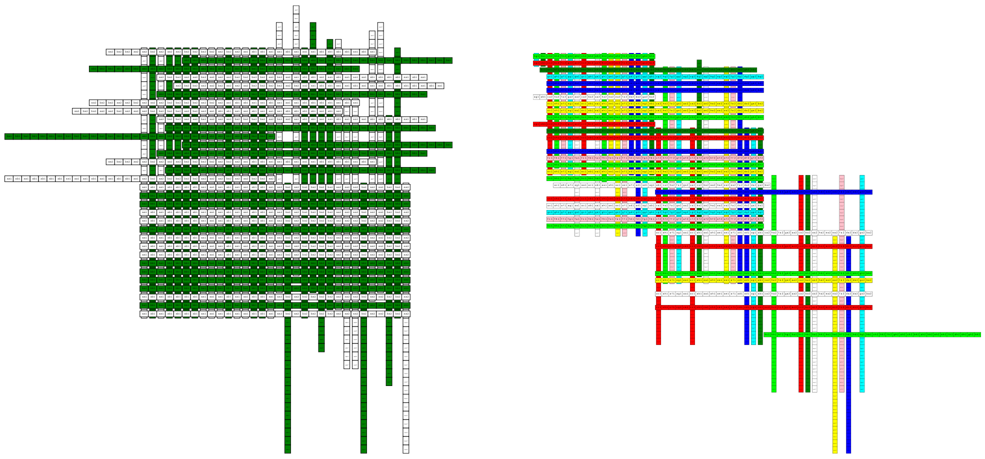


Figure 16 A typical section from ribbons with slat count 2 at $G_{mc}/G_{se} = 10$.

For both the slat count 4 and 8 ribbons, overall growth rates decrease monotonically with G_{mc}/G_{se} which is to be expected as higher values of G_{mc}/G_{se} correspond to higher temperatures. Surprisingly however, the slat count 2 ribbons seem to exhibit significantly reduced growth rates as G_{mc}/G_{se} decreases below about 9.5. Given that this region corresponds to growth with a significant amount of errors, this suggests, that the erroneous slat attachments eventually accumulate to the point where no further attachments are possible. Indeed, when we look at the assemblies resulting from these simulations, we find that this is the case as depicted in Figure 17a. This behavior is unique to the slat count 2 ribbons; for ribbons with slat counts 4 and 8, erroneous attachment below the point where errors become common rarely seems to result in stalled growth and instead typically results in uncontrolled growth as depicted in Figure 17b. This discrepancy between slat count 2 ribbons and those using slat counts of 4 or 8 can be explained by considering the point at which errors become common. For slat count 2, errors become common below $G_{mc}/G_{se} = 9$. At this ratio, slats still need several bound glues to attach stably and since growth is uncontrolled it's likely that all sites in which slats could attach stably quickly fill up. Compare this to the ribbons with slat counts 4 and 8 where errors are only common when G_{mc}/G_{se} is relatively small. Since only a few matching glues are required for stable attachment at these values of G_{mc}/G_{se}, it's much more likely that even during uncontrolled growth, there will be numerous sites in which slats can attach stably.

7.2 Details regarding ribbon systems with fixed total slat concentrations

Here we provide a few additional details regarding what we observed in the simulations of ribbons with varying amount of slat reuse in Section 4.4. Interestingly, the error rates for these



(a) A typical assembly from the simulation of our ribbons with slat count 2 at $G_{mc}/G_{se} = 7$. Here, errors have accumulated and filled in most sites surrounding the assembly which could admit stable attachments.

(b) A typical section from the growing edge of our simulated ribbons with slat count 8 at $G_{mc}/G_{se} = 2.5$. Here errors are plentiful and growth is uncontrolled.

Figure 17 Example assemblies from simulations of two different ribbon systems with cooperativity 16.

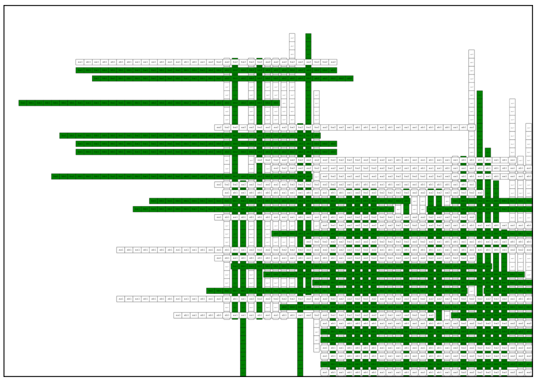


Figure 18 The growth front of a typical ribbon using slat count 2 with total slat concentration fixed at 480nM.

ribbons didn't seem to change significantly with the adjusted concentrations. Conversely, as should be expected, growth rates increased as slat counts decreased. These results agree well with the observations from [28] and suggest that by reducing slat counts further should be a viable approach to designing slat systems which grow more quickly. Suprisingly, unlike with the ribbons keeping individual slat counts fixed, growth of the ribbons using slat count 2 did not drop significantly when G_{mc}/G_{se} dropped below about 9. With individual slat concentrations fixed, the slat count 2 ribbons, reached a point where no further stable tile attachments were possible, however when total slat concentrations were fixed across ribbons, the increase to the individual slat concentrations allowed continued growth despite erroneous attachments being dominant. Figure 18 illustrates the growth front of such a ribbon when total slat concentrations were fixed which can be contrasted with Figure 17a for fixed individual slat counts.

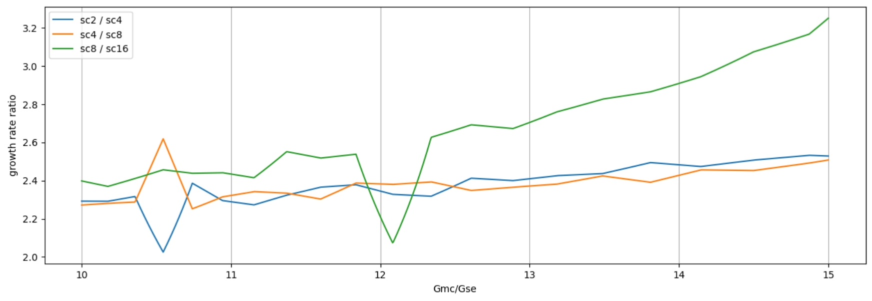


Figure 19 Ratios between growth rates for ribbons with slat counts differing by a factor of 2. Notice that this ratio never drops below 2 in the illustrated range, representing ideal conditions for error-free growth.

7.3 The Sierpinski triangle in tiles and slats

An example of a portion of the Sierpinski triangle self-assembled from aTAM tiles is shown in Figure 20, and (one layer of) our implementation with slats using cooperativity 4 is shown in Figure 22. This construction uses a gadget for each tile type from Figure 20, which can be seen in Figure 21.

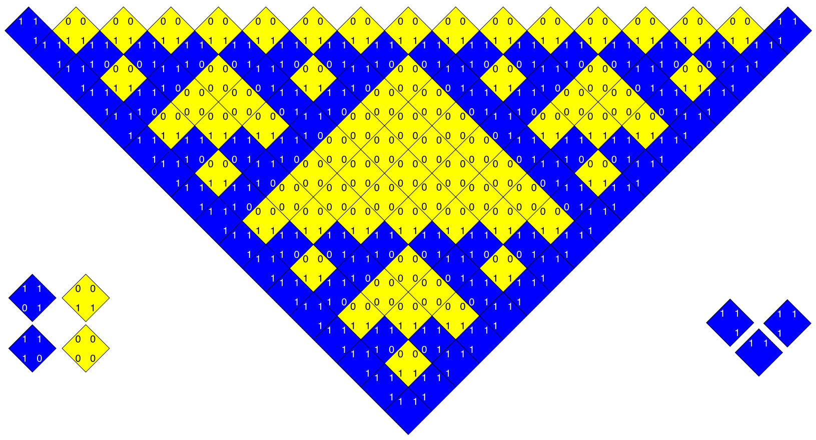


Figure 20 The Sierpinski triangle pattern. (left) The four "logic" tiles which have their inputs on the bottom and outputs on the top. The output bits are the logical "exclusive or" (`xor`) of the input bits, (middle) A portion of the infinite assembly, (right) The boundary tiles.

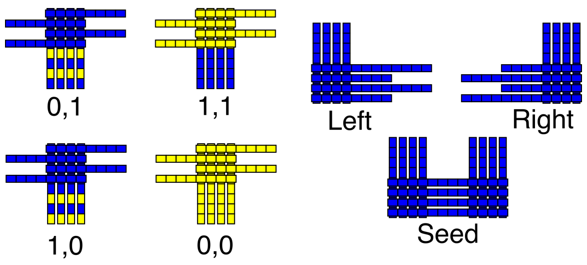


Figure 21 The gadgets of the Sierpinski triangle construction with slats using cooperativity 4 analogous to the tiles of the aTAM construction in Figure 20.

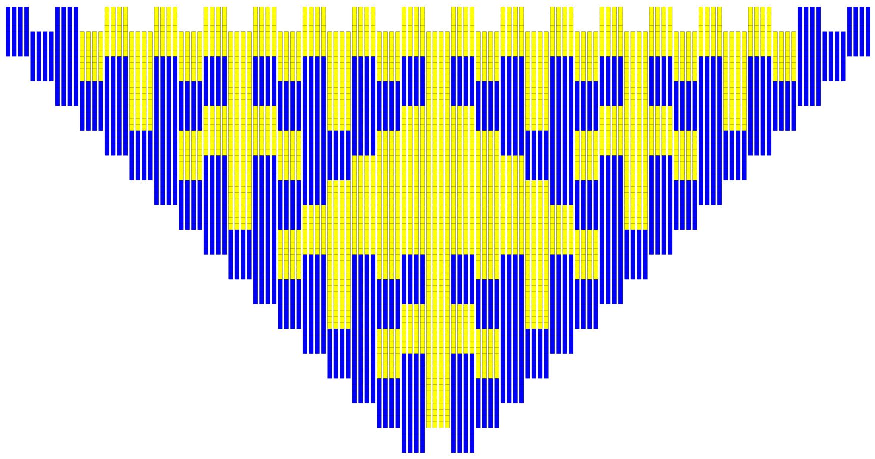


Figure 22 The Sierpinski triangle pattern made from a system of slats with cooperativity 4 (i.e. total length 8 each except for some boundary slats which are length 18). The gadgets of the construction are shown in Figure 21. Only the layer of vertical slats is shown, for clarity.

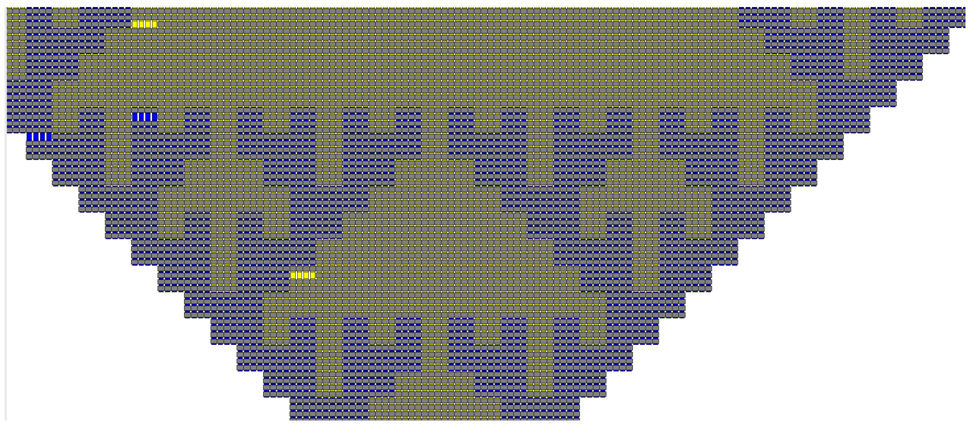


Figure 23 Growth errors during growth of a cooperativity 4 Sierpinski triangle using a slat count of 4 at $G_{mc}/G_{se} = 2.5$. Notice that some erroneous attachments are visible, leaving gaps where some horizontal slats should be, yet growth of the Sierpinski triangle pattern continues unaffected.

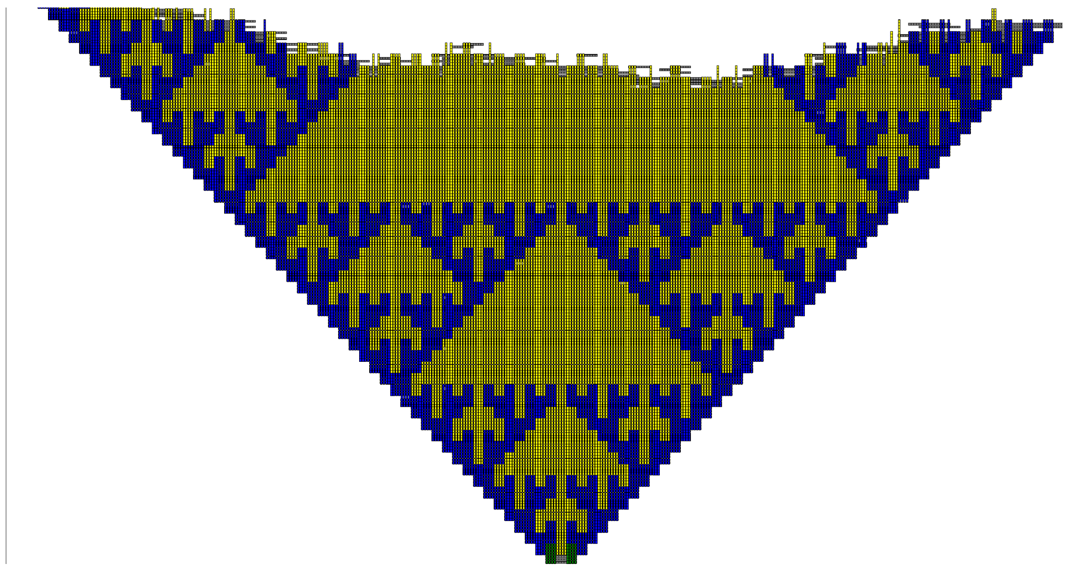


Figure 24 A zoomed out view of the complete assembly formed by the same simulation as illustrated in Figure 23. Here, horizontal slats are hidden so that the Sierpinski triangle pattern of the verical slats is more visible. This system was grown at $G_{mc}/G_{se} = 2.5$ where the probability of growth errors was distinctly non-zero. Still, the Sierpinski triangle pattern grows flawlessly.