Self-Healing Tile Sets

Erik Winfree

California Institute of Technology, Pasadena, CA 91125 winfree@caltech.edu

Summary. Molecular self-assembly appears to be a promising route to bottom-up fabrication of complex objects. Two major obstacles are how to create structures with more interesting organization than periodic or finite arrays, and how to reduce the fraction of side products and erroneous assemblies. Algorithmic self-assembly provides a theoretical model for investigating these questions: the growth of arbitrarily complex objects can be programmed into a set of Wang tiles, and their robustness to a variety of possible errors can be studied. The ability to program the tiles presents an alternative to directly physical or chemical means for reducing error rates, since redundant information can be stored so that errors can be detected, corrected, and/or prevented during the self-assembly process. Here we study the ability of algorithmic self-assembly to heal damage to a self-assembled object. We present block transforms that convert an original error-prone tile set into a new tile set that performs the same construction task (at a slightly larger scale) and also is able to heal damaged areas where many tiles have been removed from the assembly.

1 Algorithmic Crystal Growth

Biology provides the synthetic chemist with a tantalizing and frustrating challenge: to create complex objects, defined from the molecular scale up to meters, that construct themselves from elementary components, and perhaps even reproduce themselves. This is the challenge of bottom-up fabrication. The most compelling answer to this challenge was formulated in the early 1980's by Ned Seeman, who realized that the information carried by DNA strands provides a means to program molecular self-assembly, with potential applications including DNA scaffolds for crystallography [1] or for molecular electronic circuits [2]. This insight opened the doors to engineering with the rich set of phenomena available in nucleic acid chemistry [3].

This paper focuses on what might be considered the most elementary phenomenon, the self-assembly of macromolecular crystals. As commonly practiced today, DNA self-assembly is a two-stage process. In the first stage, which typically occurs at elevated temperatures, DNA oligonucleotides self-assemble

into well-defined molecular complexes often known as DNA tiles (e.g. [4]). In the second stage, which typically occurs at substantially lower temperatures, the DNA tiles stick to each other and form crystalline arrays [5]. This self-assembly is mediated by single-stranded "sticky ends" with complementary sequences that allow tiles to stick to each other by forming a double-helical domain. The situation becomes particularly interesting when there are multiple types of DNA tiles containing multiple types of sticky ends. Under such circumstances, the ground state of the crystal might not be a periodic arrangement of the molecular units [6], which motivates generalized concepts of crystalline order [7]. Since in general a physical system may take exponentially long to reach its ground state (meaning that large perfect structures will effectively never form), the use of DNA self-assembly for bottom-up fabrication requires an understanding of the kinetics of crystal growth processes – and the means to control them.

Inspired by Seeman's work, Len Adleman's work on DNA computing [8], and Hao Wang's work on the mathematical theory of tilings [9], algorithmic self-assembly [10] provides a mechanism whereby crystal growth can do information processing. A crystal of DNA tiles can store information in the spatial arrangement of the different sticky-end types exposed on its surface or along its perimeter. When a DNA tile binds to a particular sticky-end combination, it covers them up and simultaneously exposes new sticky-ends – thus effectively modifying the information presented by the crystal. A set of DNA tiles with particular input and output sticky ends therefore corresponds to a program that leaves the trace of its operations embedded in the crystal. As one application, the program can direct the construction of a shape [11]; in fact, in an "error-free" model, self-assembly is universal for the construction of arbitrary shapes [12]. From this perspective, algorithmic self-assembly presents us with an extremely simplified model of morphogenesis based on elementary crystalline growth mechanisms. The most interesting case occurs when the information present in a small "seed assembly" directs the growth of a specific shape or pattern much larger than the seed. Thus algorithmic self-assembly may be compared to biological development, a process that operates robustly over 24 orders of magnitude in volume from the information encoded in DNA to the mature organism.

For algorithmic self-assembly to direct growth at such a large scale, error-free assembly cannot be assumed and fault-tolerance becomes a central issue. Previous work suggested that physically reversible self-assembly can perform "proofreading" on redundantly-encoded information [13], that by preventing undesired nucleation on growth facets exponentially low error rates can be achieved [14], and that spontaneous nucleation of undesired assemblies unrelated to the seed can be made arbitrarily rare [15] – all with only a modest increase in the complexity of the tile set. Considered together, this work appears to solve (at least theoretically) the basic issues for fault-tolerant self-assembly according to reversible, error-prone growth processes.

Here, we consider a new model: repair of a self-assembled structure after gross damage – be it destruction by cosmic rays, fragmentation by ripping, or attack by an adversary. We call a tile set **self-healing** if, at any point during error-free growth, when any n tiles (not including the seed) are removed, subsequent error-free growth will perfectly repair the damage in average time $O(n)$. Although hints of self-healing were seen in prior work [13], large damaged areas almost always healed imperfectly. In fact, no previously considered tile sets for 2D algorithmic patterns are self-healing according to this formal definition. Nonetheless, we present a construction that transforms a tile set of interest into a self-healing tile set (containing 25 times more tile types) that performs the same assembly task but at a 5-fold larger scale. This transformation works for a wide class of original tile sets, including all widely-studied examples.

Since the self-assembled pattern was originally produced by algorithmic growth in the forward direction, the information required for repairing the hole is already present along the perimeter of the hole, and forward growth will re-build the correct structure – unless backwards growth (which is generally not guaranteed to be correct) gets there first. The key to our construction is to prevent holes (caused by damage) from filling in backwards. This is done by replacing each tile in the original tile set with a 5×5 block of tiles; however, each block is designed such that it can grow in only one direction, "forward." To achieve this, we rely on the technique developed in [14], wherein a pattern of strong bonds, weak bonds, and null bonds within each block controls the order in which tiles can be attached. A simplified 3×3 transformation (not general, but sufficient for transforming a tile set that constructs an infinite Sierpinski triangle pattern) is also shown, as well as a 7×7 transformation that has additional robustness to a type of spurious nucleation error.

1.1 Models for Algorithmic Self-Assembly

The abstract Tile Assembly Model (aTAM) provides a rigorous framework for analyzing algorithmic self-assembly. In the formulation used here, *tiles* are considered to be unit squares with each side labeled by a *bond type*. Tiles cannot be rotated. Each bond type has an associated *strength*, which may be 0, 1, or 2 (respectively called null, weak, and strong bonds). A *tile set* is a finite set of tile types, which may be used with replacement during the assembly process. Assembly begins with a specified *seed tile*. A tile may be legally added to an assembly whenever it may be placed so as to match one or more sides with a total bond strength greater than or equal to 2 (i.e., if it either forms at least two weak bonds or one strong bond). Mismatches neither help nor hinder. Assemblies that can be created from the seed tile via a sequence of legal tile additions are called the *produced* assemblies. (For a more formal description, see Ref. [12].)

Figure 1 gives three examples of algorithmic self-assembly from prior work [16, 17, 13]. The Sierpinski tile set, for example, consists of 7 tile types:

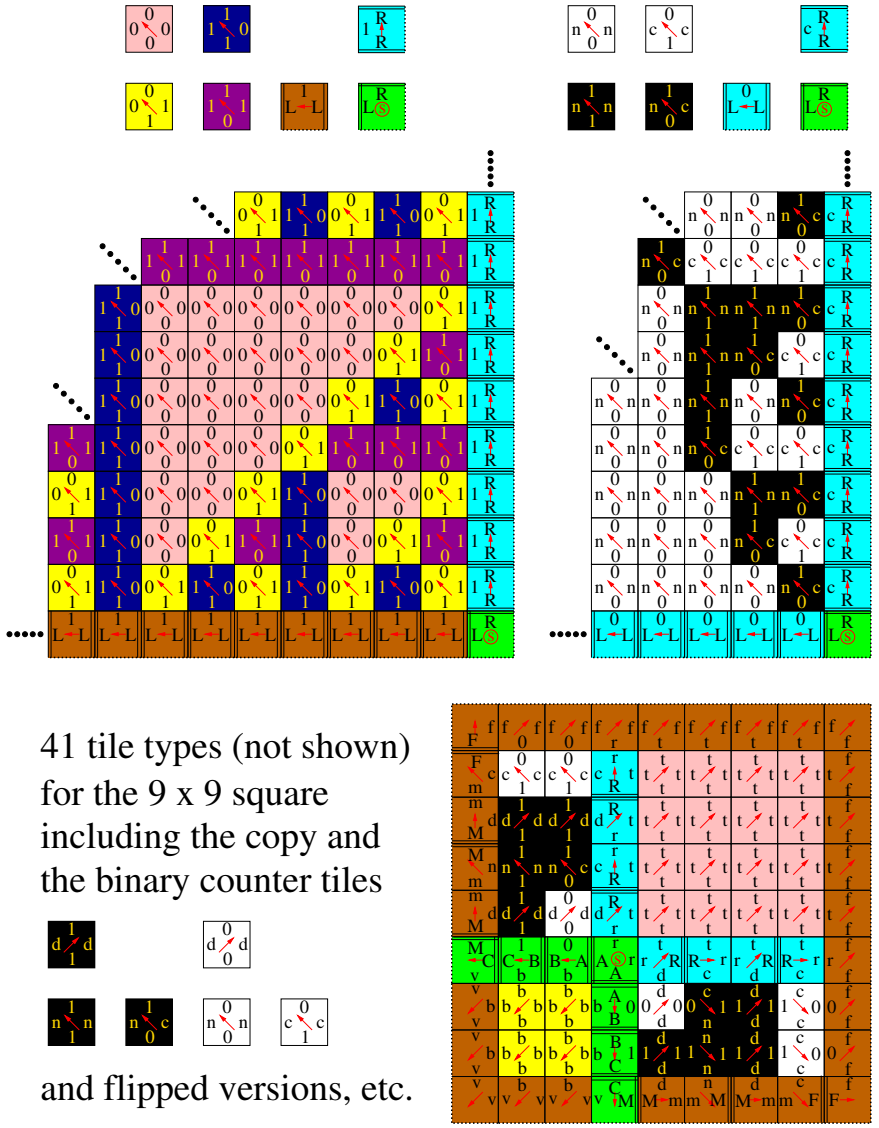

Fig. 1. Three tile sets demonstrating algorithmic growth. Bond types are indicated by letters or digits. Double lines indicate a strong bond, dotted lines indicates a null bond, all other bonds (single lines) are weak. **s** indicates the seed tile. Top left: The Sierpinski tile set. Top right: The binary counter tile set. Bottom: A tile set for constructing a 9×9 square.

four *rule tiles* (each with four weak bonds), two *boundary tiles* (each with two strong bonds, a weak bond, and a null bond), and one *seed tile* (with two strong bonds and two null bonds). Each tile type is given a distinct color.

Growth is unbounded. In the limit, the pattern formed by the pink and yellow tiles give the positions of 0's in a discrete Sierpinski fractal; the other tiles represent 1's. The red arrows, which are not part of the tiles per se, describe the forward growth process by which the assemblies were formed: diagonal arrows point away from two weak input sides (i.e., sides by which the tile attaches to the crystal), while horizontal and vertical arrows point away from a strong input side. At each forward growth site, there is a unique tile type that matches sufficiently many sides to be legally added according to the aTAM; in fact, the rule tiles implement the logic of XOR. The binary counter tile set is similar but uses different logic. Here, tile type colors are chosen to create a derivative pattern. The positions of black tiles in the n^{th} row above the blue tiles correspond to 1's in the binary expansion of the integer n. In general, by endowing each tile type with a color, the assembled tiling may produce a pattern with less complexity than the tiling itself, since the information processing required to construct the pattern is hidden. In some cases, in fact, a single color is used and one is interested only in the shape of the assembled structure. This is the case for the third tile set, which forms a finite square. (Tile types are colored just to aid understanding of the growth logic.) Here, two orthogonally-oriented binary counters count down from an initial number encoded in the green tiles; when they reach zero, the growth is terminated. The 41 tile types (not all shown) may be inferred as the set of distinct tile types that appear in the assembly. This technique can be used to construct an $N \times N$ square by replacing just the green tiles with $O(\log N)$ new tiles encoding the size of the square to be constructed.

Tile additions in the aTAM are non-deterministic, in the sense that at any given moment there are typically several locations where a tile may be legally added; for some tile sets there may also be locations at which more than one tile type may be legally added. Therefore, many tile sets will produce different assemblies dependent upon the order in which tiles are added. The three examples of Figure 1, however, uniquely produce the assemblies shown. How do we know this? Thankfully, there is a simple yet powerful technique for establishing that this is so for a tile set of interest. Consider an *assembly sequence* of legal tile additions in a particular order. For each tile, we define the *input sides* to be the sides that created weak or strong bonds when the tile was added; the *propagation sides* to be those that serve as the input sides for subsequent tile additions; and the remaining sides (if any) are called *terminal sides*. An assembly sequence is *locally deterministic* if (1) every tile addition makes *exactly* either two weak bonds or one strong bond (i.e., a strength-2 addition), and (2) if the tile at location (i, j) and all tiles abutting its propagation sides are removed from the final assembly, then there is exactly one tile type that can be legally added at (i, j). This is easy to check, tile by tile. Furthermore, if a tile set has a locally deterministic assembly sequence, then the same final assembly is produced regardless of the order in which tiles are added legally. (For a more formal description and a proof, see Theorem 2.3 of Ref. [12]. Here, we also allow infinite assembly sequences, which poses

no problems for the proof.) Locally deterministic tile sets include the majority of examples considered in the literature[1]. Furthermore, the definitions of self-healing tile sets and transformable tile sets introduced later in this paper use ideas similar to local determinism.

The aTAM is considered an "error-free" model because perfect assembly can be guaranteed, despite the asynchronous and non-deterministic order of tile additions. This is the appropriate level of abstraction for reasoning about how to program algorithmic self-assembly. However, considering how algorithmic growth can occur in a physical system, such as DNA tiles in solution, requires more realistic models that admit a variety of error modes expected to be present in any real chemistry. For example, the kinetic Tile Assembly Model (kTAM) describes physically reversible assembly as a continuous-time markov process in which tiles may be added at a location at a rate proportional to their concentration ($k_f = k[\text{tile type}] = ke^{-G_{mc}}$) regardless of how well they match their neighbors, but tiles also fall off at a rate determined by the total strength, b, of bonds holding them to their neighbors ($k_{r,b} = ke^{-bG_{se}}$). Thus, tile additions that are illegal in the aTAM will sometimes occur in the kTAM and may persist due to the addition of subsequent tiles that stabilize them – resulting in assemblies containing errors. However, if $G_{mc} \approx 2G_{se}$, then exactly the legal tile additions have $k_f \geq k_{r,b}$ (favorable growth) while exactly the illegal tile additions have $k_{r,b} \geq k_f$ (unfavorable growth). Thus, error rates can be reduced to arbitrarily low values by simultaneously decreasing tile concentration and the temperature [16]. This improvement in fidelity comes at the expense of speed: an m-fold reduction of errors requires m^2-slower growth conditions.

Errors can be reduced dramatically without significant slow-down using the technique of block transforms of tile sets that increase their robustness [13]. The transformed tile set contains more tile types but produces the same pattern as the original tile set, although at a larger scale[2]. The basic principle is to make assembly steps cooperative, so that multiple mistakes must occur before erroneous information can be used in subsequent steps; in physically reversible assembly, this gives the erroneous tiles ample opportunity to dissociate before becoming embedded in the crystal – a simple form of "proofreading". Specifically, robustness is achieved by using redundant or distributed information encoded in the bond types and by controlling the growth path by clever placement of strong bonds and null bonds [14]. These techniques can produce tile sets that are robust to several distinct types of errors that can occur in the kTAM: *growth errors* in which a weakly-binding tile attaches at a location where another tile could and should have been added; *facet nucleation errors* in which a weakly-binding tile attaches at a location where no tile should yet

[1] Not all tile sets that uniquely produce an assembly are locally deterministic; it is a sufficient but not necessary condition.

[2] Transformations that don't increase the scale also exist, but they come at the cost of a dramatic increase in the number of tile types for most patterns [18, 19].

be added; and *spontaneous nucleation errors* in which a large assembly grows in the absence of a seed tile. By replacing each tile in the original tile set by a $k \times k$ block of new tiles, growth and facet nucleation errors [14] can be reduced exponentially (in k) with only moderate slow-down. A similar exponential reduction can be achieved for nucleation errors in a mass-action variant of the kTAM [15]. Each of these works addresses only certain error types and provides a construction that works for a limited class of tile sets. Therefore, the outstanding issue for fault-tolerant algorithmic self-assembly according to reversible, error-prone growth processes is whether these methods can be combined into a single transformation that works for a wide class of tile sets and simultaneously solves all three types of errors. Although we do not yet have a definitive answer to this question, it appears that the basic principles have been identified and the foundation has been laid for a complete solution.

1.2 The challenge of self-healing crystals

Here we consider a qualitatively new type of error: gross damage to an assembly, such as a puncture, that removes a region containing many tiles. Such events are so rare in the kTAM as to be effectively non-existent, yet it is easy to imagine physical circumstances that would result in gross damage, such as fragmentation and ripping induced by fluid flow or interaction with other objects in solution. The question is whether an algorithmic crystal subject to such misfortune will be capable of healing the damage correctly. This self-healing behavior was observed to occur frequently, but not always, in kTAM simulations of proofreading tile sets [13]. Can self-healing behavior be *guaranteed* for some tile sets? We formulate this question with respect to the aTAM, so as to focus on the information-propagation aspects of the problem rather than on the probabilistic aspects.

Definition 1. *We call a tile system* **self-healing** *(in the aTAM) if the following property holds for any produced assembly: If any number of tiles are removed such that all remaining tiles are still connected to the seed tile, then subsequent growth is guaranteed to eventual restore every removed tile without error.*

Several questions come to mind. First, why the restriction that remaining tile are still connected to the seed? If gross damage breaks an assembly into several pieces, we might wish that *all* fragments regrow properly. But some fragments could be very small – just a few tiles – and it is unreasonable to expect correct regrowth in all such cases. On the other hand, since we know that growth from the seed tile is capable of constructing the entire assembly, it is also capable of *re*-constructing it, at least *if* tile additions occur in the right order. Rather than attempt to discern exactly which fragments can support regrowth, we will be satisfied with just the seed fragment[3]. So, are our favorite

[3] Salamanders can regrow their tails, but their tails can't regrow the salamander.

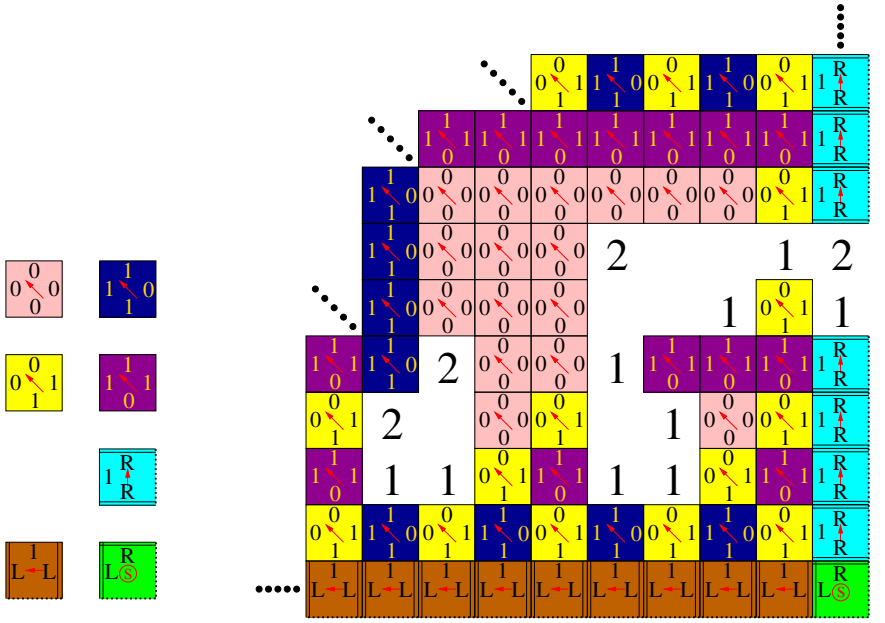

Fig. 2. Erroneous re-growth of a Sierpinski assembly after damage that removed $n = 19$ tiles. Numbers at empty sites indicate the number of distinct tile types that could be legally added at a site according to aTAM growth; where more than one tile may be added at a location, correct regrowth is no longer guaranteed. Note that the positions where incorrect regrowth can occur are dependent upon the tile set; the binary counter tiles, for example, exhibit non-deterministic regrowth in different growth directions (regrowth from north and west inputs is sometimes ambiguous, as is regrowth from east and west inputs) than the Sierpinski tiles (regrowth from north and west inputs is always ambiguous).

tile sets self-healing? This can be quickly answered, in the negative, for the three tile sets shown in Figure 1. Several types of erroneous regrowth are shown for the Sierpinski tile set in Figure 2; similar errors occur in the other tile sets. Then do any self-healing tile sets exist? Yes; the simplest example is a periodic crystal in which every bond type is unique to the two tile types it joins, and all bonds are strong. Uniquely-addressed finite assemblies can also be self-healing. This is, however, an extremely limited class of self-assembled patterns. Can algorithmic self-assembly be self-healing? It is far from obvious.

A first hope might be that robustness-enhancing tile set transformations, such as the original [13] and snaked [14] proofreading schemes, already provide self-healing properties. While kTAM simulations do show improved ability to regrow into punctures, it is not perfect, and in the aTAM errors are even more frequent. Examination of those block transformations suggests that typically both proofreading approaches will result in new tile sets that suffer the same regrowth problems as the original tile sets.

The remainder of this paper shows that self-healing is possible for algorithmic self-assembly. We first present a 3×3 block transformation that can be applied to tile sets, like the Sierpinski and binary counter tile sets, that grow within a quarter-plane from a L-shaped boundary. A proof technique is developed for showing that the resulting tile sets are indeed self-healing. The simplicity of these techniques makes it straightforward, then, to design and test block transformations that work for a wider class of tile sets. We present a 5×5 scheme that works for many (though not all) locally deterministic tile sets, including all three examples from Figure 1. Finally, we ask how these results are affected if regrowth occurs not one tile at a time (as in the aTAM), but by the addition of strongly-connected chunks of tiles that may have formed on their own without the seed (which we call the polyomino aTAM). Under these more challenging conditions, self-healing is still possible, but our construction uses 7×7 blocks. We conclude with a discussion of open questions.

2 Self-healing transformations for quarter-plane patterns

What makes self-healing hard? The problem is that whereas the original tile set may have been deterministic when growing in the expected directions, with the expected input sides and propagation sides, regrowth may occur in any direction and tile additions may no longer be deterministic. For example, in the case of quarter-plane growth from a L-shaped boundary, the rule tiles have four weak bonds, two of which serve as input and two as output; while there must be a unique tile for any input pair, there may be multiple tiles that have the same output pair. If such a tile is removed, the other (incorrect) tile could be added during regrowth, binding by the two weak bonds on its output sides. Incorrect regrowth could also occur if two tiles share some combination of an output side and an input side (either adjacent or opposing). Between the Sierpinski tile set and the binary counter tile set, all such situations occur.

Quarter-plane growth from a L-shaped boundary is a rich class of tile sets, capable of creating a great variety of patterns. In fact, it is sufficient for universal computation by simulation of blocked cellular automata or Turing machines [10, 16]. In general we may wish to use rule tiles simulating a blocked cellular automaton that outputs $< f(x,y), g(x,y) >$ for input $< x, y >$ where x and y are from some possibly large finite alphabet and $f(\cdot,\cdot)$ and $g(\cdot,\cdot)$ are arbitrary functions[4]. Input is provided by the boundary tiles that create the L; locally deterministic growth allows each arm of the L to consist of a finite initial sequence of boundary tiles followed by a finite repeating sequence. We

[4] Reversible 1D cellular automata, for which the "inputs" $< x, y >$ are also a function of the "outputs" $< f, g >$, are a widely-studied class that includes Turing-universal computation [20]. Tile sets directly simulating these cellular automata would be immune from ambiguity in the backward regrowth direction, but the possibility for problems in other directions remains.

need a block transformation that works for all such tile sets, which we call L-BCA tile sets[5].

There are two options: either to make sure that in the transformed tile set sufficient information is present for any direction of regrowth, or else to ensure that regrowth in the wrong direction is impossible. The former seeming impossible, we take the second approach. The only way to prevent backward and sideways regrowth is for the transformed tile set to contain null bonds at key positions that control the growth path; the principle here is adapted from the mechanism that prevents facet nucleation in the snaked proofreading construction [14]. A 3×3 self-healing block transformation is shown in Figure 3, wherein each original tile produces 9 new tiles with labels and bond strengths according to a template that depends upon the original tile's bond strength pattern. (Rotated tiles use rotated templates.) For each tile type $t =< a, b, c, d >$ in the original tile set, nine tile types are included in the new tile set. The new bond types are indexed variants of the original tile type (e.g. $t3$) or bond type (e.g. $a5$). These are respectively called *tile-type bonds* and *bond-type bonds*. New tiles that use at least one tile-type bond are given the same color as the original tile and are called *block tiles*, whereas new tiles that use exclusively bond-type bonds are left uncolored and are called *bond tiles*. The same bond tile type may result from the transformation of distinct original tiles. Examination of a damaged crystal grown from the transformed Sierpinski tile set illustrates the inability to grow backwards or sideways where there is any potential ambiguity. But how can we prove that this is always the case? We use two simple lemmas, stated informally but hopefully unambiguously.

Lemma 1. *If a tile can be added at a particular site in some assembly, then it can be added at the same site (if it is open) in any larger assembly that contains all the same tiles (and then some).*

This follows immediately from the threshold condition for tile addition in the aTAM: bond strengths are non-negative and mismatches do not interfere (i.e., they contribute strength zero). □

Lemma 2. *Consider an assembly produced from a tile set according to the aTAM. Remove any single tile (not the seed), as a test. The test succeeds if there is a unique tile that can now be added at that site according to aTAM growth. The tile set is self-healing if and only if this test succeeds for every possible tile in every possible produced (i.e., correct) assembly.*

First, the easy implication is immediate: if a test fails, then the tile set is not self-healing. For the converse, now suppose that a tile set is not self-healing. That means that there is some pattern of damage, and some sequence of regrowth that leads to a first incorrect tile t. Prior to adding t, every tile that

[5] These are quarter-plane tile sets discussed in Ref. [19], but augmented by a seed tile and the boundary tiles.

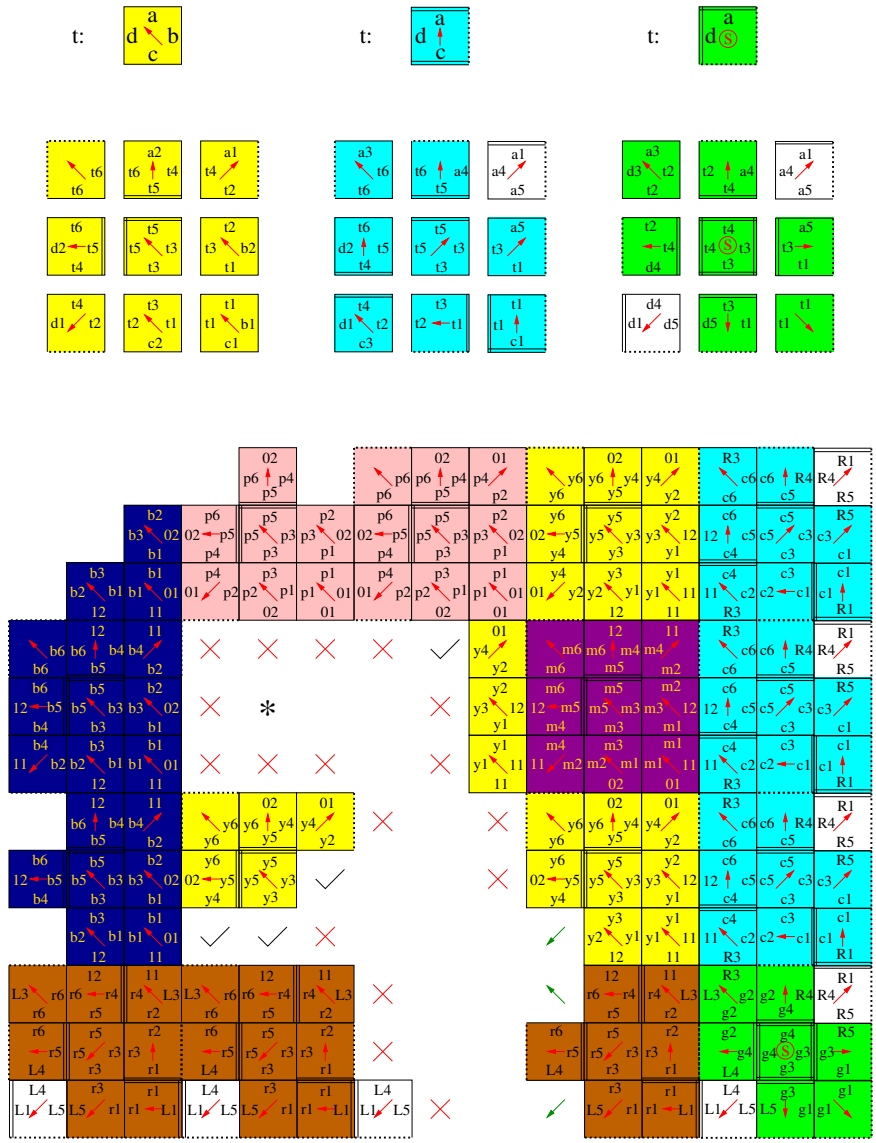

Fig. 3. Top: Templates for the 3×3 self-healing transformation for rule tiles, boundary tiles, and the seed tile. Bottom: A damaged assembly grown using the transformed Sierpinski tile set. Sites may allow no immediate regrowth (red crosses), regrowth from incorrect input sides (black checks), or regrowth from correct input sides (green arrows). Note that a series of legal tile additions allows the partial yellow blocks to regrow independent of other activity, but the upper left damaged block (*) cannot regrow until both its input blocks have formed.

was present was correct. Add to this assembly all the other tiles (other than at t's site) that had been removed in the damage. We now have a produced (correct) assembly with a single tile removed. By Lemma 1, t can be added in this assembly too. So can the correct tile, which is different from t. We have thus identified a test that fails. □

We can now prove that the 3×3 self-healing transformation works for all L-BCA tile sets.

Theorem 1. *The 3×3 block transformation shown in Figure 3 produces a self-healing tile set when applied to any L-BCA tile set. Furthermore, the resulting tile set will construct the same pattern as the original tile set, but at a 3-fold larger scale; specifically, the majority color of each block will be identical to the corresponding tile in the original pattern.*

To prove this, we need to show (1) that aTAM growth from the seed will produce the correct pattern (at a larger scale), and (2) that every test conceivable by Lemma 2 is bound to succeed. For claim (1), all we need to do is identify a locally deterministic assembly sequence. This is easy, since all L-BCA tile sets are locally deterministic, we can start with a locally deterministic assembly sequence for the original tile set, and show that we can elaborate it into an assembly sequence for the transformed tile set that remains locally deterministic. For each tile added in the original sequence, we add a series of 9 tiles for the corresponding block of the transformed tile set. Since we know which sides are the input sides for the original L-BCA tile (south and east for rule tiles, south or east for boundary tiles, and no inputs for the seed tile), it is easy to find a canonical series of tile additions for each transformed block, assuming the blocks for the corresponding inputs are already completely present. Therefore, in the blocks each tile has a canonical growth direction (as illustrated) and it can easily be verified that each tile addition is locally deterministic (exactly strength-2, and growth from input and terminal sides is unique). When at least one input side is within the block, uniqueness is automatically guaranteed; when binding via a single strong bond as the first tile in a boundary block, uniqueness follows from the uniqueness of tile addition on the boundary in the original tile set; when binding via two weak bonds as the first tile in a rule block, uniqueness follows again from uniqueness of tiles with a given input pair in the original tile set. Thus, we have constructed a locally deterministic assembly sequence for the transformed tile set. This establishes part (1) of the result.

Part (2) can also be established by local examination of the transformed blocks, using Lemma 2. For each tile within each block, we examine all possible combinations of sides that contain total bond strength at least 2, and we ask whether there is a unique tile that matches those sides. For rule blocks, eight tiles need at least one side internal to the block, which therefore establishes uniqueness; the exception is the tile in the lower right corner, which can grow from input sides on the south and east – but for L-BCA tile sets, there is a unique tile with this pair of inputs. So no test can fail within a rule block.

For boundary blocks, there are three exceptions to the rule that at least one side must be internal to the block; these are (a) the bond tile, whose sides are unique to that bond type; (b) the lower right tile, which is unique because boundary growth in the original tile set is unique; and (c) the lower left tile, which is unique for the same reason. For the seed block, the only exceptions are again the bond tiles (which are unique) and the upper left tile (which is the only tile touching both a horizontal and a vertical boundary, and thus is unique). This establishes the conditions for Lemma 2, and thus completes the proof that the transformed tile set is indeed self-healing. □

This proof also helps us understand why it was necessary to include "bond tiles", which at first seem like an out-of-place hack: if both boundary blocks and the seed block had block-specific tiles in the upper right corner, then backward growth from a boundary block (with a damaged region underneath it) would no longer be unique – sometimes the seed block corner tile would attach in this position. In fact, we will see that bond tiles play an important role in the more general self-healing transformations to come.

3 A general self-healing transformation

We now know that it is possible to have algorithmic growth that is self-healing. Unfortunately, L-BCA tile sets, though computationally universal, do not include most examples of algorithmically-generated morphology, such as the square of Figure 1, which exhibit much greater variety in the growth path. We would therefore like a self-healing transformation that will work for any locally deterministic tile set, thus being applicable to essentially all algorithmic self-assembly tile sets considered in the literature (e.g. [11, 21, 22, 12]). However, two situations are allowed in locally deterministic tile sets that cause technical difficulties for the block transformations that are presented below: first, the same tile type might appear with different input sides and propagation sides at different locations within an assembly; and second, the correct final assembly might contain weak bonds along the outer perimeter or internal mismatches between tiles. Rather than attempt to handle these possibilities, we choose to restrict our attention to tile sets in which neither of these situations arises. Many tile sets in the literature are already of this form, and others can be converted with a little thought. In any case, we are lead to the following definition.

Definition 2. *A* **transformable** *tile set is a locally deterministic tile set with the additional properties that (1) each tile type always appears with the same sides as input, propagation, and terminal sides, and (2) all non-null bonds are either input sides or propagation sides.*

This means that all final structures are "capped" by tiles with null bonds on the outside, as is the case in the square assembly of Figure 1. It also means

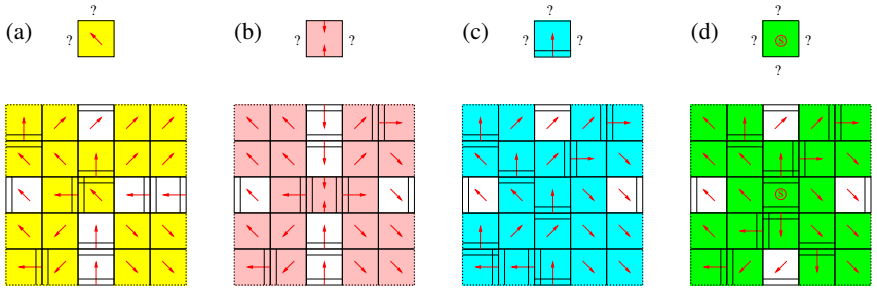

Fig. 4. A 5×5 self-healing transformation. Top: The four bond-strength patterns for tile input sides. Non-input sides (indicated by ?'s) may have any strength. Bottom: The corresponding block templates. Colored tiles are called *block tiles*; bonds between block tiles are tile-type bonds. Uncolored tiles are *bond tiles* and have exclusively bond-type bonds. Within each block template, each bond type appears uniquely (in any given direction). Bond-type bonds in equivalent positions of different blocks are of the same type. If the original tile had a null bond, then the corresponding bond tile in the template is replaced by a block tile with 3 tile-type bonds and a null bond. Original tiles with rotated input bond patterns use rotated templates.

that tile types can be labeled with arrows indicating their input sides, and these labels are correct for all locally deterministic assembly sequences.

Rather than the three bond-strength patterns (rule tiles, boundary tiles, seed tiles) of L-BCA tile sets, transformable tile sets may have a great variety of bond-strength patterns. We can avoid having a separate block transform for every bond-strength pattern by adopting a uniform convention for the interface between blocks that is the same regardless of whether a strong bond or a weak bond is being represented. By using the same block transform for all tile types with the same input bond-strengths pattern (regardless of the strengths on the non-input sides), we require only four block schemes to be specified, as shown in Figure 4 for a 5×5 self-healing transformation. The four cases are (a) *diagonal rule blocks*, in which two adjacent weak bonds serve as the input; (b) *convergent rule blocks*, in which two weak bonds on opposite sides of the tile serve as the input; (c) *strong blocks*, in which a single strong bond serves as the input; and (d) *seed blocks*, which have no input. Note that bond tiles now play a much more prominent role in the blocks. The growth pattern within each block is designed so that output bond tiles receive their input in a clockwise growth direction; this way, rotated blocks that define the same bond tile will use it with a consistent growth direction.

Theorem 2. *The 5×5 block transformation shown in Figure 4 produces a self-healing tile set when applied to any transformable tile set. Furthermore, the resulting tile set will construct the same pattern as the original tile set, but at a 5-fold larger scale; specifically, the majority color of each block will be identical to the corresponding tile in the original pattern.*

The proof of this theorem follows exactly along the lines of the proof for the 3×3 transformation, but with more cases to test for Lemma 2. Tests of block tiles succeed because either at least one input is another block tile from the same block, or else all inputs (from bond tiles) specify a unique tile due to the original tile set being locally deterministic. Tests of bond tiles succeed because all their bonds are unique to the particular bond tile type. □

This transformation is now sufficient for showing that the growth of arbitrary algorithmic shapes [12] can be self-healing. Incidentally, the transformation of a transformable tile set is also a transformable tile set, an interesting closure property that could aid in combining different robustness transformations.

We can now revisit the question of why bond tiles were necessary. The essential reason is that in the original tile set, strong bonds dictate a deterministic tile choice in the forward growth direction, but may be non-deterministic in the backward growth direction. This difficulty is compounded by our choice (made for the convenience of being able to write the block transformation concisely) to treat output sides uniformly for both weak and strong outputs. Consequently, *every* output side has a strong bond, and non-deterministic backward growth could be severe. Thankfully, by padding all sides of the block with null bonds, we can prevent the backward growth from continuing for more than a single tile – the bond tile. However, all those null bonds make forward growth difficult for diagonal blocks and convergent blocks, because the two pieces of information required to know the new block's type are not co-localized. The solution in this case is to project that information into the center of the tile by a non-committal growth process (bond tiles); the actual decision is then made in the center where the information can be combined.

4 Self-healing for polyomino tile sets

Tile sets produced by the 5×5 self-healing transformation have a lot of strong bonds, even when the original tile set had quite few. This elicits some concern from those familiar with physical self-assembly, because it brings into question the assumption that growth occurs only from the seed tile, and that all subsequent steps consist of the accretion of a single isolated tile at a time, rather than by the aggregation of separately-nucleated fragments. In the absence of the seed tile (for the seed block), one can consider aTAM growth from each of the other tiles in the tile set. Ideally, such growth cannot proceed far, thus supporting the accretion hypothesis in spirit if not in detail. However, we are not so lucky with this 5×5 transformation. The worst offenders here are the strong blocks: starting with first tile in the block's usual assembly sequence as a "mock seed", aTAM growth puts together the entire 25-tile block, and possibly more. This is just asking for trouble.

We therefore consider whether it is possible to create self-healing tile sets in which significant spurious nucleation does not occur, and for which aggre-

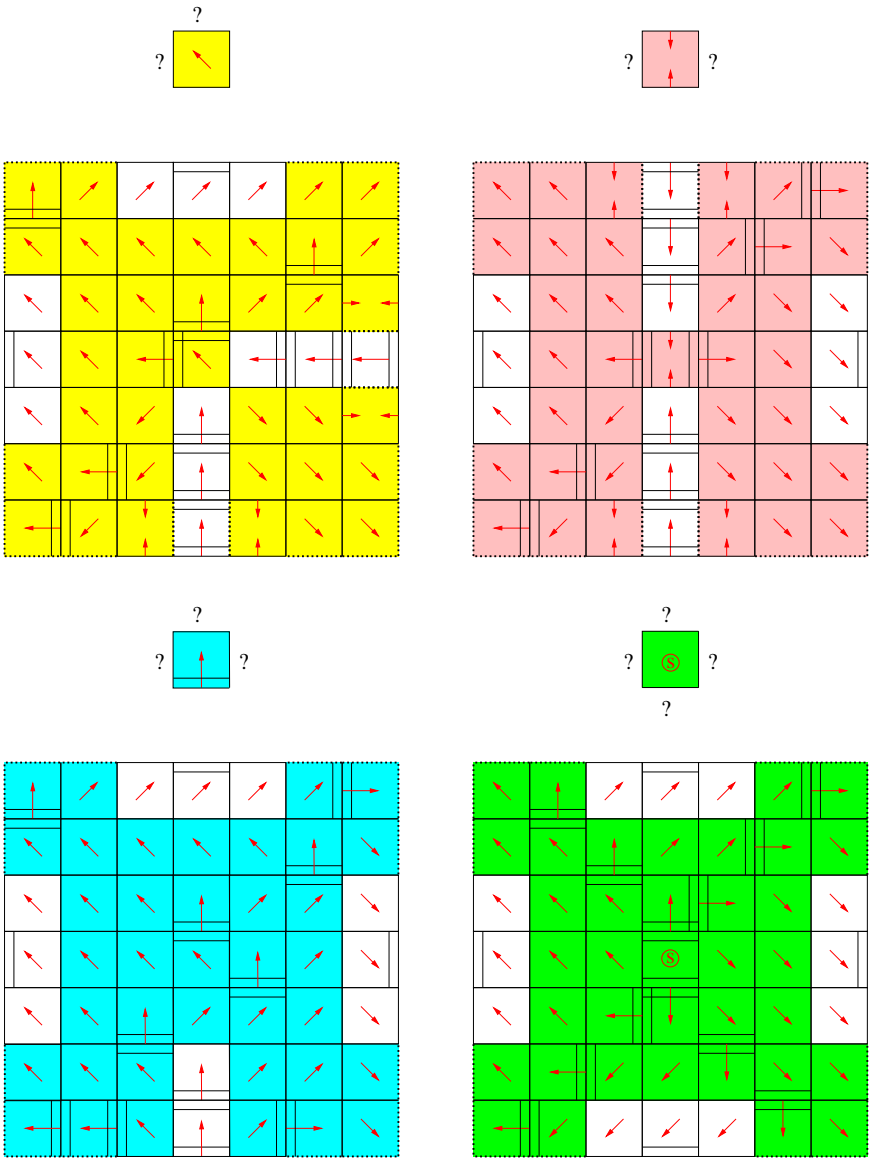

Fig. 5. A 7×7 self-healing transformation that yields polyomino-safe tile sets. Top: The four bond-strength patterns for tile input sides. Bottom: The corresponding block templates. Note that each side of each block now exposes one strong bond and two weak bonds.

gation of seeded assemblies with spuriously nucleated assemblies is too weak to proceed, except when it results in correct assemblies. Previous work on controlling spurious nucleation in a mass-action kTAM model made use of

the principle that growth from a non-seed tile must take several unfavorable steps (which would not be allowed in the aTAM) before unbounded favorable growth (allowed in the aTAM) becomes possible [15]. Essentially, the solution presented there corresponds to a block transformation in which strong bonds are placed sufficiently far apart; in fact, instead of using tiles with strong bonds, in that work such tiles were permanently stuck together and treated as a single polyomino tile with each unit side containing a weak bond (or a null bond). The polyomino formalism provides a suitable "worst-case" framework for treating aggregation. (Our model is essentially the same as the "multiple tile" model of Ref. [23].)

Given a tile set that uniquely produces a target assembly under aTAM growth from the seed, we will define a corresponding set of polyominoes. Begin with the given tile set excluding the seed tile – this is the first step in the construction of all possible spuriously nucleated assemblies (here called polyominoes). Now iterate: if it is possible to place two such assemblies next to each other such that they can form bonds with a total strength at least 2, then add the resulting assembly to the set of polyominoes. If this process does not terminate or if any polyomino is not a subset of the target assembly, declare failure; the given tile set is not polyomino-safe. Otherwise, we have a finite set of polyominoes representing assemblies that have spuriously nucleated and aggregated without the seed tile.

The polyomino aTAM begins with the seed tile and allows the addition of any polyomino (in the set defined above) placed such that it can form bonds with a total strength of at least 2. Under the polyomino aTAM, any assembly that was produced by the aTAM can still be produced, since all individual tiles are also in the polyomino set (except for the seed tile itself, which is not used for growth in transformable tile sets). Possibly additional (and thus incorrect) assemblies can also be formed when polyominoes are used. For our purposes, uniqueness will follow from the polyomino-safe self-healing property – if deviations from the correct tile placement is impossible during re-growth, then it must also have been impossible during growth the first time around.

Definition 3. *We say a tile set gives rise to* **polyomino-safe self-healing** *if the following property holds for any produced assembly: If any number of tiles are removed such that all remaining tiles are still connected to the seed tile, then subsequent growth according to the polyomino aTAM with the corresponding polyomino set is guaranteed to eventually restore every removed tile without error.*

To prove that a tile set has this property, we need polyomino variants of the previous lemmas.

Lemma 3. *If a polyomino can be added at a particular site in some assembly, then it can be added at the same site (if it is open) in any larger assembly that contains all the same tiles (and then some).*

Lemma 4. *Consider an assembly produced from a tile set according to the aTAM. Choose a polyomino from the corresponding polyomino set, and choose a location where it overlaps existing tiles. (It necessarily does not overlap the seed tile.) As a test, remove all overlapped tiles. The test succeeds if either the polyomino makes no more than a single weak bond with the remaining assembly, or if all tiles in the polyomino are identical with the removed tiles. The tile set gives rise to polyomino-safe self-healing if and only if this test succeeds for every possible case.*

The proofs are straightforward adaptations of the proofs of the previous lemmas. □

It now becomes straightforward, although tedious, to verify the following:

Theorem 3. *The 7×7 block transformation shown in Figure 5 produces a polyomino-safe self-healing tile set when applied to any transformable tile set. Furthermore, the resulting tile set will construct the same pattern as the original tile set, but at a 7-fold larger scale; specifically, the majority color of each block will be identical to the corresponding tile in the original pattern.*

The corresponding polyomino set contains only small polyominoes (no more than 4 tiles each) that consist of either entirely bond tiles or entirely block tiles. Bond polyominoes can only replace identical bond tiles, since their bond-type bonds are unique. Block polyominoes may have both tile-type bonds and bond-type bonds. Most block polyominoes have no more than one bond-type bond; therefore, to attach, the polyomino must make at least one tile-type bond, which uniquely positions it within the correct block. The only exceptions occur at the centers of diagonal and convergent rule blocks and at the input to strong blocks. At these sites, a block polyomino may bind by bond-type bonds with strength 2, but in these cases uniqueness is guaranteed by the original tile set being locally deterministic. □

This tile set operates on the same principles as the 5×5 tile sets, with the added precaution that in order for a strong block to grow, the central strong bond tile must be supported by tiles presenting weak bonds on either side. By distributing responsibility for propagating information through the sides of the blocks, no single tile on its own is capable of nucleating the growth of the entire block. Note that even if the original tile set was *not* polyomino-safe, the transformed tile set will be.

5 Open questions

We now know that self-healing is possible in passive self-assembly. How good can it get?

Generality and optimality of the block transformations. The first question is whether a wider class than the "transformable" tile sets can be made self-healing. Tile sets that produce a language of shapes – rather than uniquely producing a target assembly – are clearly not going to work, because self-healing can't be guaranteed at the first non-deterministic site. But might it be possible to find a transformation that works for any locally deterministic tile set?

Scale is an important issue for self-assembled objects [12, 18]. In previous work on fault-tolerant self-assembly (in the kTAM), increased robustness was achieved at the cost of increased scale [14, 15, 19]. In this work (in the aTAM), a maximal level of robustness is achieved with a constant scale-up – seven-fold, for polyomino-safe self-healing. It is intriguing to ask whether the strategies of Refs. [18, 19] can be use to produce that self-healing tile sets that incur *no* scale-up costs – although this will come at the cost of an increase in the number of tile types. The technical challenge, in this case, concerns the bond tiles, which will not necessarily carry the color of the block they appear in.

Can self-healing be achieved without the use of extra strong bonds and null bonds, which presumably make a self-assembled molecular object more fragile? In this case, most tiles will be rule tiles (i.e., they will have four weak bonds), and therefore a puncture will be able to grow back in from any direction. The self-healing property requires, in this case, that no two rule tiles may have any pair of identically-labeled sides. This seems very restrictive. How restrictive?

We chose here to define "self-healing" with respect to the fragment of a damaged assembly that contains the seed tile – we were not concerned with what happens when the other fragments regrow. In fact, there are some situations, such as when just a small region containing the seed is destroyed, for which it would be very desirable if regrowth could repair the damage. This seems in principle possible for some definition of "small", for example by having unique bonds in a region surrounding the seed. How can this robustness be quantified, and can a general construction be found that achieves arbitrary levels of robustness for a small cost?

Robustness to continual damage. So far, we have considered repairing an isolated damage event, and we have shown that it is possible to do so. What if there is repeated damage, with punctures of various sizes occurring at various rates? If the damage events are sufficiently far apart in space and time, then each puncture will be completely healed before any further damage occurs nearby. The expected time to repair n-tile damage is $O(n)$, since in the worst case there is a linear chain of dependencies and the n sites must be filled in that order. Thus, even if damage events have a weak power law distribution (i.e., with a long tail), self-healing tile sets should be able to maintain the correct pattern: we have a guarantee that any tile added to the assembly will be correct, and the only question is whether tiles are being removed faster or being replaced faster. Figure 6 shows simulations that confirm this intuition, in a variant of the aTAM in which each tile type is tested to be added at each site with forward rate f (as a continuous-time markov process) [21].

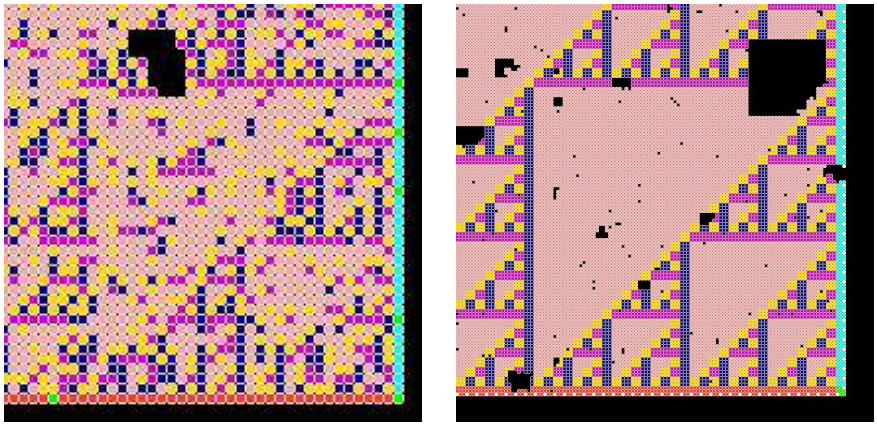

Fig. 6. Growth under a barrage of damage events. Size $k \times k$ square puncture events occur (centered at any given tile) at a rate $1000k^4$-fold less than the forward rate f for tile addition. (I.e., an exactly 10×10 hole will be punctured somewhere in a 100×100 area in about the same time as it takes for 1000 tiles to visit a particular site and attempt to bind. There being 61 tile types in the assembly on the right, this corresponds to about once every 17 successful tile additions, i.e., 17 layers of tiles regrown.) Left: The original Sierpinski tile set. The target Sierpinski pattern has not yet been entirely erased and can still be discerned. Right: The 3×3 transformed tile set. The scale is reduced by a factor of 3, so that each 3×3 block is the same size as a single original tile on the left. The simulation was allowed to run four times as long (in terms of events per tile). Except for holes that are in the process of healing, the entire Sierpinski pattern is perfectly correct.

However, there is a catch. Two catches. The first is that for many natural models of environmental damage, the distribution of event sizes has *very* long tails. This is due to the connectivity constraint: damaging or removing a small number of tiles from an assembly may result in a disconnected fragment, and thus necessitate the formal removal of a large number of additional tiles. This is particularly severe in long thin assemblies and near the corner of L-shaped assemblies. The second catch is that there is a finite rate at which either the seed tile itself will be destroyed, or barring that, a small region around the seed tile will be disconnected from the rest of the assembly. This means that every so often, the entire structure will have to regrow from the seed – a hard reboot. Is it possible that algorithmic growth can be designed to repair itself even when a region containing the seed tile is removed?

Performance in the kTAM. At the beginning of this paper, we mentioned earlier work that addressed how to make a tile set more robust to growth errors, facet nucleation errors, and spurious nucleation errors in physically reversible models such as the kTAM. Here, we examined robustness to punctures – which seems like an error mode orthogonal to the previously examined ones – and analyzed how to achieve robustness in the aTAM, so as to focus

on the new aspects of this problem. How well do our solutions work in the kTAM? Preliminary tests with the 3×3 self-healing tile set show that although it is a great improvement over the original 1×1 tile set, it does not perform dramatically better than the simpler 3×3 proofreading tile set of Ref. [13]. We can attribute this to two factors: first, the self-healing tile set uses only two sides of each block to encode information – rather than all three in the proofreading tile set – and therefore it suffers a higher rate of growth errors. Secondly, even when proofreading tiles regrow incorrectly, the growth usually does not proceed far before an inconsistency prevents further growth; this tends to stall the regrowth and allows the incorrect tiles to fall off... often, but not always. Can better performance be achieved by explicitly incorporating principles for all previously examined types of errors into the design of a block transformation that yields tile sets robust to all error types?

Experimental practicality. The study of fault-tolerant tile sets is motivated in large part by the promise of using algorithmic self-assembly for bottom-up fabrication of complex molecular devices. Theory, however, naturally leads in directions appreciated only by theorists. How practical are the self-healing tile sets presented here? For comparison, there is already on-going experimental work investigating 2×2 proofreading systems as well as 2×6 blocks for controlling spurious nucleation. Therefore, 3×3 blocks could in principle be investigated in the near future – but I would think it would be a challenging experiment! For DNA tile self-assembly, having a polyomino-safe tile set may be important to help prevent spurious nucleation, but 7×7 blocks (49-fold more tiles!) don't engender enthusiasm. Finding smaller self-healing tile sets would be a considerable advance.

A completely different approach to self-healing would be to use more sophisticated molecular components. There have already been proposals for DNA tiles that reduce self-assembly errors by means of mechanical devices (implemented by DNA hybridization and branch migration) that determine when a tile is ready to attach to other tiles or when it can be replaced by other tiles [24, 25, 26]. Although intimidating to experimentally develop such a complex tile, these approaches may ultimately have great pay-off as they can in principle reduce all the types of errors discussed in this paper, and the resulting complex tiles are likely to be much smaller than the e.g. 7×7 blocks presented here.

Finally, there are more serious types of physical damage that could occur. For example, within the damaged area, some tiles might be broken such that they continue to stick to the crystal, but no further tiles can stick to them. It seems that removing such tiles would require active processes.

6 Discussion

As Ned tells it, DNA nanotechnology began with a vision of an Escher print and a scheme for creating DNA crystals using 6-armed junctions – which we

now know won't work. Nonetheless, this vision has lead to an incredible richness of experimentally-demonstrated DNA structures, devices, and systems, which confirms the validity of the original insight. This gives the theorist some hope that in this field, persistently pursuing a compelling idea can lead to something real – even if the original formulation is tragically flawed. Most importantly, Ned's vision has inspired new fields of research that seem to have taken on a life of their own.

Consider passive molecular self-assembly of the sort discussed in this paper. It is a small corner of DNA nanotechnology, devoid of complicated DNA structures, nanomechanical devices, catalysts and fuels, and other sophisticated inventions. Even so, passive self-assembly has revealed itself to be more interesting than I ever would have imagined! Rather than appearing more and more like crystals (the lifeless stuff of geology), passive self-assembly now seems a microcosmos for the fundamental principles of biology – at least, if seen through a blurry and somewhat rose-colored lens. Specifically, passive molecular self-assembly seems to encompass several of the main aspects for how molecularly-encoded information can direct the organization of matter and behavior:

Programming. How can one specify a molecular algorithm? Algorithmic self-assembly – a natural generalization of crystal growth processes – is Turing-universal [10]. The choice of a tile set is a program for self-assembly. This shows that molecularly-encoded information can be very simple (just the complementarity of binding domains) and yet capable of specifying arbitrarily complex information processing tasks.

Complexity. What kinds of structures can be self-assembled, and at what costs? In fact, any shape with a concise algorithmic description can be constructed by a concise tile set – at some increase in scale [12]. There is a single tile set that acts as a universal constructor; given a seed assembly containing a program for what shape to grow (encoded as a pattern of bond types presented on its perimeter), this tile set will follow the instructions in a way vaguely reminiscent of a biological developmental program.

Fault-tolerance. Can errors in self-assembly be reduced sufficiently to approach biological complexity? Biological organisms often grow by many orders of magnitude from their seed or egg, and often the mature individual consists of over 10^{24} macromolecules. All this despite the stochastic, reversible, and messy biochemistry underlying all the molecular processes. Reducing errors in algorithmic self-assembly to this level seems quite challenging, but theoretical constructions for error-correcting tile sets [13, 14, 15] appear to do the job – at least, on paper.

Self-healing. Can severe environmental damage be repaired? The purpose of this paper has been to show that if the damage is simply the removal of tiles in the damaged region, then it is possible to design algorithmic tile sets that heal the damage perfectly.

Self-reproduction and evolution. Can algorithmic crystals have a life cycle? The copying of genetic information from layer to layer in a crystal is a simple algorithmic task. If, when haphazardly fragmented, both pieces of the original crystal contain copies of the same information, then one can say the the information has been reproduced. If the information has a selective advantage, for example serving as the program for some algorithmic growth process, then Darwinian evolution can be expected to occur [27].

Remarkably, what seems to be the most elementary physical mechanism – crystallization – is already capable of exhibiting many of the phenomena commonly associated with life [28].

Acknowledgements. The author is indebted to discussions with Ashish Goel, Ho-Lin Chen, Rebecca Schulman, David Soloveichik, Matthew Cook, and Paul Rothemund. This work was partially funded by NSF award #0523761.

References

1. Nadrian C. Seeman. Nucleic-acid junctions and lattices. *Journal of Theoretical Biology*, 99(2):237–247, 1982.
2. Bruce H. Robinson and Nadrian C. Seeman. The design of a biochip: A self-assembling molecular-scale memory device. *Protein Engineering*, 1(4):295–300, 1987.
3. Nadrian C. Seeman and Philip S. Lukeman. Nucleic acid nanostructures: bottom-up control of geometry on the nanoscale. *Reports on Progress in Physics*, 68:237–270, 2005.
4. Tsu-Ju Fu and Nadrian C. Seeman. DNA double-crossover molecules. *Biochemistry*, 32:3211–3220, 1993.
5. Erik Winfree, Furong Liu, Lisa A. Wenzler, and Nadrian C. Seeman. Design and self-assembly of two-dimensional DNA crystals. *Nature*, 394:539–544, 1998.
6. Charles Radin. Tiling, periodicity, and crystals. *J. Math. Phys.*, 26(6):1342–1344, 1985.
7. A.L. Mackay. Generalised crystallography. *Izvj. Jugosl. centr. krist. (Zagreb)*, 10:15–36, 1975.
8. Leonard M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, November 11, 1994.
9. Hao Wang. An unsolvable problem on dominoes. Technical Report BL-30 (II-15), Harvard Computation Laboratory, 1962.
10. Erik Winfree. On the computational power of DNA annealing and ligation. In Lipton and Baum [29], pages 199–221.
11. Paul Wilhelm Karl Rothemund and Erik Winfree. The program-size complexity of self-assembled squares. In *Symposium on Theory of Computing (STOC)*. ACM, 2000.
12. David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. In Ferretti et al. [30], pages 344–354. Extended abstract; preprint of the full paper is cs.CC/0412096 on arXiv.org.

13. Erik Winfree and Renat Bekbolatov. Proofreading tile sets: Error-correction for algorithmic self-assembly. In Junghuei Chen and John Reif, editors, *DNA Computing 9*, volume LNCS 2943, pages 126–144, Berlin Heidelberg, 2004. Springer-Verlag.
14. Ho-Lin Chen and Ashish Goel. Error free self-assembly using error prone tiles. In Ferretti et al. [30], pages 62–75.
15. Rebecca Schulman and Erik Winfree. Programmable control of nucleation for algorithmic self-assembly. In Ferretti et al. [30], pages 319–328.
16. Erik Winfree. Simulations of computing by self-assembly. Technical Report CS-TR:1998.22, Caltech, 1998.
17. Erik Winfree. Algorithmic self-assembly of DNA: Theoretical motivations and 2D assembly experiments. *Journal of Biomolecular Structure & Dynamics*, pages 263–270, 2000. Special issue S2.
18. John H. Reif, Sudheer Sahu, and Peng Yin. Compact error-resilient computational DNA tiling assemblies. In Ferretti et al. [30], pages 293–307.
19. David Soloveichik and Erik Winfree. Complexity of compact proofreading for self-assembled patterns. Extended abstract to appear in *DNA Computing 11*.
20. K. Morita. Computation-universality of one-dimensional one-way reversible cellular automata. *Information Processing Letters*, 42:325–329, 1992.
21. Leonard M. Adleman, Qi Cheng, Ashish Goel, and Ming-Deh A. Huang. Running time and program size for self-assembled squares. In *ACM Symposium on Theory of Computing (STOC)*, pages 740–748, 2001.
22. Qi Cheng and Pablo Moisset de Espanes. Resolving two open problems in the self-assembly of squares. Computer science technical report #03-793, University of Southern California, 2003.
23. Gagan Aggarwal, Qi Cheng, Michael H. Goldwasser, Ming-Yang Kao, Pablo Moisset de Espanes, and Robert T. Schweller. Complexities for generalized models of self-assembly. *SIAM Journal on Computing*, 34:1493–1515, 2005.
24. A. J. Turberfield, B. Yurke, and A. P. Mills, Jr. DNA hybridization catalysts and molecular tweezers. In Erik Winfree and David K. Gifford, editors, *DNA Based Computers V*, volume 54 of *DIMACS*, Providence, RI, 2000. American Mathematical Society.
25. Ho-Lin Chen, Qi Cheng, Ashish Goel, Ming deh Huang, and Pablo Moisset de Espanés. Invadable self-assembly: Combining robustness with efficiency. *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 883–892, 2004.
26. Kenichi Fujibayashi and Satoshi Murata. A method of error suppression for self-assembling DNA tiles. In Ferretti et al. [30], pages 113–127.
27. Rebecca Schulman and Erik Winfree. Self-replication and evolution of DNA crystals. To appear in the proceedings of the *VIIIth European Conference on Artificial Life (ECAL)*.
28. A. G. Cairns-Smith. *The life puzzle: on crystals and organisms and on the possibility of a crystal as an ancestor*. Oliver and Boyd, New York, 1971.
29. Richard J. Lipton and Eric B. Baum, editors. *DNA Based Computers*, volume 27 of *DIMACS*, Providence, RI, 1996. American Mathematical Society.
30. Claudio Ferretti, Giancarlo Mauri, and Claudio Zandron, editors. *DNA Computing 10*, volume LNCS 3384, Berlin Heidelberg, 2005. Springer-Verlag.