BOUNDS ON THE TILE COMPLEXITY OF SHAPES IN SELF-ASSEMBLY SYSTEMS

by

Christine Stoll
B.Sc., Simon Fraser University, 2006

A Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science
in the Department
of
Mathematics

SIMON FRASER UNIVERSITY
Summer 2009

APPROVAL

Name: Christine Stoll

Degree: Master of Science

Title of Thesis: Bounds on the Tile Complexity of Shapes in Self-Assembly Systems

Examining Committee: Dr. Ralf Wittenberg
Chair

Dr. Ladislav Stacho
Senior Supervisor

Dr. Ján Maňuch
Co-Supervisor

Dr. Petr Lisoněk
Supervisor

Dr. Jason Bell
SFU Examiner

Date Approved: July 10, 2009

Declaration of
Partial Copyright Licence

Abstract

In this thesis we study self-assembly systems, in particular, we focus on the tile complexity of shapes. Using the standard tile assembly model proposed by Rothemund and Winfree, we give two lower bounds on the tile complexity of arbitrary shapes in terms of the radius and diameter of the shape. Applying our results to a square yields a partial answer to a problem of Rothemund and Winfree. We also introduce a new model of self assembly—the step assembly model—which can significantly reduce the number of tile types needed to assemble a given shape. For this model, we give an upper bound on the tile complexity of arbitrary shapes and exhibit a family of shapes with constant tile complexity. Furthermore, we relate the tile complexity of a shape in the step assembly model to the well known node search number of its underlying spanning tree.

Keywords: self-assembly; tile systems; complexity of shapes; DNA nanotechnology

Acknowledgments

I would like to thank my supervisor, Ladislav Stacho, and my co-supervisor, Ján Maňuch, for all their support. I would also like to thank Petr Lisoněk and Jason Bell for their comments on improving this thesis.
Danke an meine Familie, die trotz großer Entfernung immer für mich da ist.

Contents

List of Figures

Chapter 1

Introduction

Self-assembly is the process by which simple parts autonomously assemble into larger, more complex objects. Self-assembly occurs in nature, for example, when atoms combine to form molecules, and molecules combine to form crystals. It has been suggested that intricate self-assembly schemes will ultimately be useful for circuit fabrication, nano-robotics, DNA computing, and amorphous computing [25, 17, 9, 1]. Research is also being conducted to self-assemble sieves to remove viruses from serum, and to nanomanufacture drug-delivery and medical-imaging devices [7].

The standard model to study the process of self-assembly is the Tile Assembly Model proposed by Rothemund and Winfree [18] (we will refer to this model as the standard tile assembly model) which considers the assembly of square blocks called "tiles" and a set of glues called "binding domains". Each of the four sides of a tile can have a glue on it that determines interactions with neighbouring tiles. It is assumed that there is an infinite supply of tiles of each tile type. The process of self-assembly is initiated by a single seed tile and proceeds by attaching tiles one by one. A tile can only bind to the growing complex if it binds strongly enough, as determined by the *temperature* τ.

Branched DNA molecules [21] provide a direct physical motivation for this model. DNA double-crossover molecules, each bearing four "sticky ends" analogous to the four sides of a tile, have been designed to self-assemble into a periodic two-dimensional lattice [24, 14, 12, 20]. The binding interactions between double-crossover molecules may be redesigned by changing the base sequence of their sticky ends, thus allowing arbitrary sets

of tiles to be investigated in the laboratory. Tiles can also be implemented using protein-based designs, where unit-length nanorods (made of proteins) are joined at right angles at their midpoints to form a plus sign [7]. Protein nanorod structures are, unlike DNA based assemblies, very rigid. It is believed that this rigidity will allow them to be used for the nanomanufacture of macroscale objects.

In this thesis we are interested in the "tile complexity" of shapes that arise from the self-assembly process. Roughly speaking, the tile complexity of a shape is the smallest number of distinct tile types required to uniquely assemble the shape. In the remainder of this chapter, we give a formal description of the standard tile assembly model and survey some tile complexity results in the standard tile assembly model. We then describe several variations of the standard model that have been proposed in an effort to reduce tile complexity and survey tile complexity results for each of these models.

In Chapter 2, we return to the standard tile assembly model and give two lower bounds on the tile complexity of arbitrary shapes. The first bound is based on the radius of the shape and the second bound is based on the Manhattan diameter of the shape. We use the second result to provide partial answer to an open problem of Rothemund and Winfree [18]. They conjectured that uniquely assembling an $N \times N$ square at temperature 1 (where we do not require a bond between every two adjacent tiles) requires at least $2N - 1$ distinct tile types. We show that this is indeed so, under the assumption that the square assembles uniquely without any binding domain mismatches. An extended abstract of the results in Chapter 2 appeared in [16].

In Chapter 3, we propose our own variation of the standard tile assembly model, called the step assembly model. This model differs from the standard tile assembly model in that a sequence of tiles sets (rather than just a single tile set) is used. We immerse a seed tile into the first tile set, "filter out" the assembled shape from this set and place this assembled shape into next tile set where it now acts as a seed and assembly continues. This process is repeated until the whole sequence of tile sets have been applied. The step assembly has the potential to significantly reduce the tile complexity of shapes. For instance, to assemble an $N \times N$ square (where there is a bond between every two adjacent tiles) N^2 distinct tile types are required in the standard model at temperature 1, whereas in the step assembly

model at temperature 1 only 9 tile types suffice. Using the step assembly model, we also provide a construction which gives an upper bound on the number of tile types required to uniquely assemble an arbitrary shape, in terms of the depth of a caterpillar decomposition of the shape. Furthermore, we show how this depth is related to the node search number of the underlying spanning tree. Lastly, we exhibit a large class of shapes (each shape obtained from another shape by scaling by a factor of two belongs to this class) that has constant tile complexity (24 tile types suffice). Furthermore, each shape scaled by a factor of two, can be assembled using at most 14 tile types.

1.1 Definition of the Standard Tile Assembly Model

We will consider the square lattice, i.e., the graph with vertex set $\mathbb{Z} \times \mathbb{Z}$ and edge set $\{uv : |u, v| = 1\}$, where $|u, v|$ denotes the distance between u and v. The directions $\mathcal{D} = \{N, E, S, W\}$ are used to indicate the natural directions in the lattice. Formally, they are functions from $\mathbb{Z}\times\mathbb{Z}$ to $\mathbb{Z}\times\mathbb{Z}$: $N(x, y) = (x, y+1)$, $E(x, y) = (x+1, y)$, $S(x, y) = (x, y-1)$, and $W(x, y) = (x - 1, y)$. Note that $E^{-1} = W$ and $N^{-1} = S$.

A *tile* is a square with the north, east, south, and west edges labeled from some alphabet Σ of *binding domains* (glues). Formally, a tile t is a 4-tuple $(t_N, t_E, t_S, t_W) \in \Sigma^4$, indicating the binding domains on the north, east, south, and west side, respectively. Note that tiles are oriented, so a rotated version of a tile is considered to be a different tile. We will use *null* to indicate the lack of a binding domain, and will assume $null \in \Sigma$. The special tile $empty = (null, null, null, null)$ represents an empty space when placed onto the grid. A *configuration* on a set of tiles T is a map $C : \mathbb{Z} \times \mathbb{Z} \to T$. We define the *vertex set* of configuration C as $V(C) = \{(x, y) : C(x, y) \neq empty\}$. A configuration C is *finite* if $V(C)$ is finite. We will refer to $C(x, y)$ as the tile at the vertex (x, y) in C. Given a configuration C and a set of vertices $V \subseteq \mathbb{Z} \times \mathbb{Z}$, a *sub-configuration* of C induced by V is the map $C[V] : \mathbb{Z}\times\mathbb{Z} \to T$ such that $C[V](x, y) = C(x, y)$ for all $(x, y) \in V$, and $C[V](x, y) = empty$, otherwise. If G is any subgraph of the lattice graph, then we sometimes abuse the notation of $C[V(G)]$ to simply $C[G]$. Given two configurations C and D, we define their union to be the following map from $\mathbb{Z} \times \mathbb{Z}$ to $T \cup \{\infty\}$:

$$(C \cup D)(x,y) = \begin{cases} C(x,y) & \text{if } D(x,y) = empty \\ & \text{or } C(x,y) = D(x,y), \\ D(x,y) & \text{if } C(x,y) = empty \\ & \text{or } C(x,y) = D(x,y), \\ \infty & \text{otherwise.} \end{cases}$$

Note that $C \cup D$ is a configuration whenever it is a map to T. Equivalently, $C \cup D$ is not a configuration if there exists $(x,y) \in V(C) \cap V(D)$ such that $C(x,y) \neq D(x,y)$.

A function $g : \Sigma \times \Sigma \to \mathbb{N} = \{0,1,2,\ldots\}$ satisfying $g(\sigma,\sigma') = g(\sigma',\sigma)$ and $g(null,\sigma) = 0$ for all $\sigma,\sigma' \in \Sigma$ is called a *strength function*. Strength functions measure the interaction strength between binding domains. In the standard tile assembly model strength functions are restricted to strength functions that satisfy $g(\sigma,\sigma') = 0$, whenever $\sigma \neq \sigma'$.

Given a tile t, a configuration C, and a direction d, we denote the interaction strength in configuration C between tile t at position (x,y) and its respective neighbouring tile by

$$g_d^C(t,x,y) = g(t_d, C(d(x,y))_{d^{-1}}).$$

Note that we do not require that $C(x,y) = t$. In particular, if $C(x,y) \neq t$, then $g_d^C(t,x,y)$, $d \in \mathcal{D}$ tells us how t would bind if it were in C. Given $(x,y) \in \mathbb{Z} \times \mathbb{Z}$ and $d \in \mathcal{D}$, we say that there is a *bond* between positions (x,y) and $d(x,y)$ in C if $g_d^C(C(x,y),x,y) \geq 1$ (in the standard tile assembly model this implies that the binding domain on the abutting sides of the two tiles is the same).

Under the standard tile assembly model a *tile system* is a 5-tuple $\mathbf{T} = (\Sigma, T, S, g, \tau)$, where T is a finite set of tiles with binding domains from Σ and contains the tile *empty*, S is a configuration on T called *seed configuration*, g is a strength function, and τ is a threshold parameter called *temperature*. Unless otherwise noted, we will be working with seed configurations consisting of a single tile; formally a configuration C_t, where $t \in T$, satisfying $C_t(0,0) = t$, and $C_t(x,y) = empty$ for all $(x,y) \in \mathbb{Z} \times \mathbb{Z} \backslash \{(0,0)\}$.

Self-assembly is now defined as a relation between configurations on T. Let C and D be

two configurations of T, such that $C = D$ except at position (x, y), where $C(x, y) = empty$, and $D(x, y) = t$, for some $t \in T \backslash \{empty\}$. Then we write $C \rightarrow_{\mathbf{T}} D$, if

$$\sum_{d \in \mathcal{D}} g_d^C(t, x, y) \geq \tau.$$

This means that a tile can be added to a configuration at position (x, y), if and only if the sum of the interaction strengths of t with its neighbours reaches or exceeds τ. The relation $\rightarrow_{\mathbf{T}}^+$ is the transitive closure of $\rightarrow_{\mathbf{T}}$.

We are interested in a subclass of configurations that arise from the self-assembly process. A tile system $\mathbf{T}$ and the relation $\rightarrow_{\mathbf{T}}^+$ define the partially ordered set of configurations called *assemblies* of $\mathbf{T}$: $Asmb(\mathbf{T}) = \{A : S \rightarrow_{\mathbf{T}}^+ A\}$, and the set of *terminal assemblies* of $\mathbf{T}$: $Term(\mathbf{T}) = \{A \in Asmb(\mathbf{T}) : \nexists B \text{ such that } A \rightarrow_{\mathbf{T}}^+ B\}$. A tile system *uniquely produces* A if for all $B \in Asmb(\mathbf{T})$, such that $B \neq A$, $B \rightarrow_{\mathbf{T}}^+ A$ (which implies $Term(\mathbf{T}) = \{A\}$). An assembly A is said to have *no binding domain mismatches* if for any two neighbouring positions (x, y) and $d(x, y)$, $d \in \mathcal{D}$, such that $A(x, y) \neq empty$ and $A(d(x, y)) \neq empty$, we have $A(x, y)_d = A(d(x, y))_{d^{-1}}$. The uniqueness assumption on the tile system has an interesting consequence:

Observation 1. *Let $\mathbf{T}$ be a tile system that uniquely produces a terminal assembly U. Let A and B be two assemblies of $\mathbf{T}$, then $A \cup B$ is an assembly of $\mathbf{T}$ as well.*

A *shape* S is a connected subgraph of the lattice induced by $V(S) \subseteq \mathbb{Z} \times \mathbb{Z}$. In particular, a shape S is an $N \times N$ *square* if there exists a position (x_0, y_0) such that $(x, y) \in V(S)$ if and only if $x_0 \leq x < x_0 + N$ and $y_0 \leq y < y_0 + N$. We say a configuration A *has shape* S, if $V(A) = V(S)$. We say a tile system *uniquely produces shape* S, if the terminal assembly of the tile system is unique and has shape S. We say an assembly A is *full* if for any two neighbouring positions (x, y) and $d(x, y)$ in $V(A)$, where $d \in \mathcal{D}$, there is a bond between them in A. We say a tile system *uniquely produces a full shape* S, if the terminal assembly of the tile system is unique, full, and has shape S.

Next we define the tile complexity of a shape, which is the focus of this thesis. The *tile complexity of a (full) shape* S is the minimum number of distinct non-*empty* tiles required

in tile system that uniquely produces the (full) shape S under the standard tile assembly model. Later on we will consider other models of self-assembly. The tile complexity for each of these models is defined analogously.

Given a standard tile system $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$, and a configuration C of T, the *backbone graph* of C, $G(C) = (V, E)$ is the subgraph of the square lattice whose vertex set is $V(C)$, and two vertices (x, y), (x', y') form an edge if and only if there is a bond between (x, y) and (x', y') in C. Note that if the configuration C is also an assembly of $\mathbf{T}$, then its backbone graph is connected. If G is a subgraph of a backbone graph we say a tile $t \in T$ *appears* on G, if there is a vertex $(x, y) \in V(G)$ such that $C(x, y) = t$.

A *translation ϕ mapping (x_0, y_0) to (x_1, y_1)* is a mapping from $\mathbb{Z} \times \mathbb{Z}$ to $\mathbb{Z} \times \mathbb{Z}$ such that a point $p = (x, y)$ is mapped to $\phi(p) = (x + (x_1 - x_0), y + (y_1 - y_0))$. The composition of $n \geq 0$ copies of a translation ϕ will be denoted as $\phi^{(n)}$. Similarly, $\phi^{(-n)}$ denotes the composition of $n \geq 0$ copies of the inverse translation ϕ^{-1}.

1.2 Tile Complexity in the Standard Tile Assembly Model

Most tile complexity results in the standard tile assembly model are limited to squares. We begin with some results of Rothemund and Winfree, who considered the number of distinct tile types needed to uniquely self-assemble squares under the standard tile assembly model at temperatures 1 and 2. Most tile complexity results, whether for the standard tile assembly model or variations thereof, consider primarily temperatures 1 and 2. Experimentally, temperature 1 conditions seem relatively easy to achieve, while temperature 2 conditions appear to be difficult to create. Furthermore, self-assembly proceeds more slowly at temperature 2. However, as we will see, assembling squares at temperature 1 requires far more tile types than at temperature 2, and in practice only a limited number of tile types can be created. Moreover, real self-assembly systems are believed to have temperature between $\tau = 1$ and $\tau = 2$ [19].

Theorem 1. [18] The tile complexity of an $N \times N$ full square at temperature 1 is N^2.

Sketch of Proof. To show that N^2 tile types suffice, construct N^2 distinct tiles, one for each position in the square, with a unique binding domain for each adjacent pair of tiles.

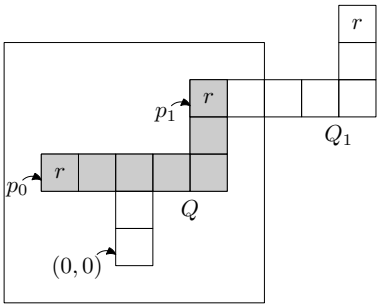


Figure 1.1: The assembly $A[R] \cup A_1[Q_1]$.

To show that N^2 tiles types are required, suppose, towards a contradiction, that there exists a tile system $\mathbf{T}$ such that the unique terminal assembly A is a full $N \times N$ square, and $\mathbf{T}$ uses fewer than N^2 distinct non-*empty* tile types. Then there must be two distinct positions, $p_0 = (x_0, y_0)$ and $p_1 = (x_1, y_1)$, such that $A(p_0) = A(p_1) = r$. By swapping coordinates if necessary, we may assume that $x_0 < x_1$. Let ϕ be the translation mapping p_0 to p_1. Let Q be the "L" shaped (or possibly linear) path consisting only of the tiles at $(x_0, y_0), \ldots, (x_1, y_0), \ldots, (x_1, y_1)$ (see Figure 1.1). For every integer n, let $Q_n = \phi^{(n)}(Q)$, i.e., the vertex p of Q corresponds to the vertex $\phi^{(n)}(p)$ of Q_n.

Let R be the tree consisting of Q and a shortest path from $(0, 0)$ (the position of the seed tile) to Q. Recall that $A[R]$ is the subconfiguration of A induced by R. Since R contains $(0, 0)$, $A[R] \in Asmb(\mathbf{T})$. We will extend this assembly by forming the union of $A[R]$ and either $A[Q_1]$ or $A[Q_{-1}]$, depending on the position of the seed tile. More precisely, choose $A[Q_1]$ if $x_1 > 0$, and choose $A[Q_{-1}]$ if $x_1 \leq 0$. For the remainder of the argument, suppose $x_1 > 0$ (an analogous argument applies if $x_1 \leq 0$). Then $A[R] \cup A[Q_1]$ is a configuration of T. Moreover, since the vertex sets of R and Q_1 intersect in x_1, the backbone graph of $A[R] \cup A[Q_1]$ is connected. Hence $A[R] \cup A[Q_1] \in Asmb(\mathbf{T})$. This can be extended indefinitely. More precisely, let $B_n = \bigcup_{i=1}^{n} A[Q_i]$. Then, for every integer $n \geq 0$, $A[R] \cup B_n \in Asmb(\mathbf{T})$. This contradicts that $\mathbf{T}$ uniquely assembles an $N \times N$ square. Thus, the minimum number of distinct non-*empty* tiles required to uniquely assemble an $N \times N$ full square is N^2. □

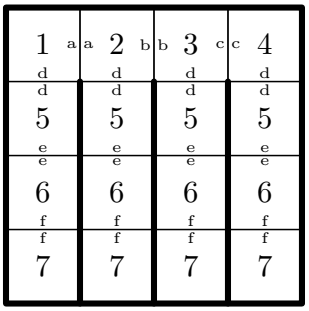


Figure 1.2: A comb-like construction of a square at temperature 1. Thick edges indicate sides with the *null* binding domain.

If we do not require the square to be full, fewer tile types suffice. In particular Rothemund and Winfree in [18] gave a comb-like construction using $2N - 1$ non-empty tile types (see Figure 1.2). They conjectured that this construction is optimal, i.e. that $2N - 1$ non-empty tile types are required to uniquely assemble an $N \times N$ square at temperature 1. We will give a partial answer to this question in Chapter 2, where we prove this claim with the additional assumption that the square assembles without any binding domain mismatches (any two adjacent tiles either form a bond or else both touching sides have the binding domain *null* assigned).

Theorem 1 completely answers the question of the tile complexity of squares at temperature 1. At temperature 2 the situation is markedly different. Self-assembly of full squares at temperature 2 requires far less tile types than at temperature 1. However, while Theorem 1 is an upper and lower bound on the tile complexity that holds for all values of N, such results do not exist for squares at temperature 2.

Theorem 2. [18] The tile complexity of an $N \times N$ full square at temperature 2 is at most $N + 4$.

Sketch of Proof. Figure 1.3 shows a construction for uniquely assembling an $N \times N$ full square using $N + 4$ non-empty tile types. Self-assembly starts from the seed tile labeled "1" in the figure and initially proceeds via strength 2 interactions to form the top boundary of the square. To continue to build the second row, tile A then attaches via its strength 2 binding domain to this boundary. This allows the remainder of the row to be assembled.

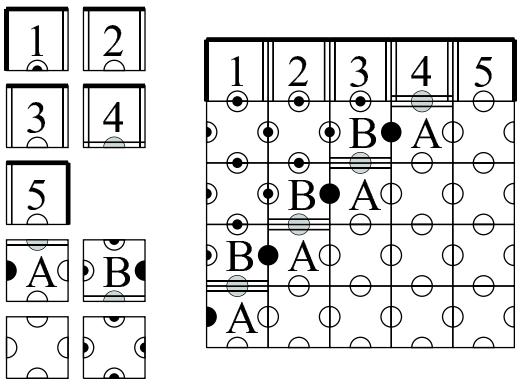


Figure 1.3: A construction for uniquely assembling an $N \times N$ square using $N+4$ non-empty tile types. Thick sides have strength 0, thin sides have strength 1, and double-lined sides have strength 2. Adjacent pairs of tiles in the top row share a unique binding domain. Source: [18].

Next, the tile A attaches via its strength 2 binding domain to the tile B to start the assembly of another row. Since with each row, tile A and tile B move one position to the left, this assembly terminates, when tile A reaches the left boundary. A single strength 1 bond is not sufficient for tile B to attach, thus the self-assembly process terminates. □

An even better bound can be obtained by combining the above construction with a fixed width binary counter. As the binary counter technique is common in this area, we decided to include the sketch of the proof.

Theorem 3. [18] The tile complexity of an $N \times N$ full square at temperature 2 is at most $\log\lceil N \rceil + 22$.

Sketch of Proof. Figure 1.4 shows a construction for uniquely assembling an $N \times N$ full square using $\log\lceil N \rceil + 22$ non-empty tile types. Let $n = \log\lceil N \rceil$, and let $c = 1 + 2^{n-1} - \lceil (N-n)/2 \rceil$. First an $(n-1) \times (n-1)$ full square is assembled using the construction in Theorem 2. However, instead of labeling the tiles of the top row by $1, 2, \ldots, n$, we now label them with the integer $c-1$ in binary (using one digit for each tile). In addition, each of these tiles has either a 0 or 1 as binding domains on their north side (both of these binding domains have strength 1). If $N-n$ is even, the binding domains are chosen such that they encode the integer c in binary. If $N-n$ is odd, the binding domains encode the integer

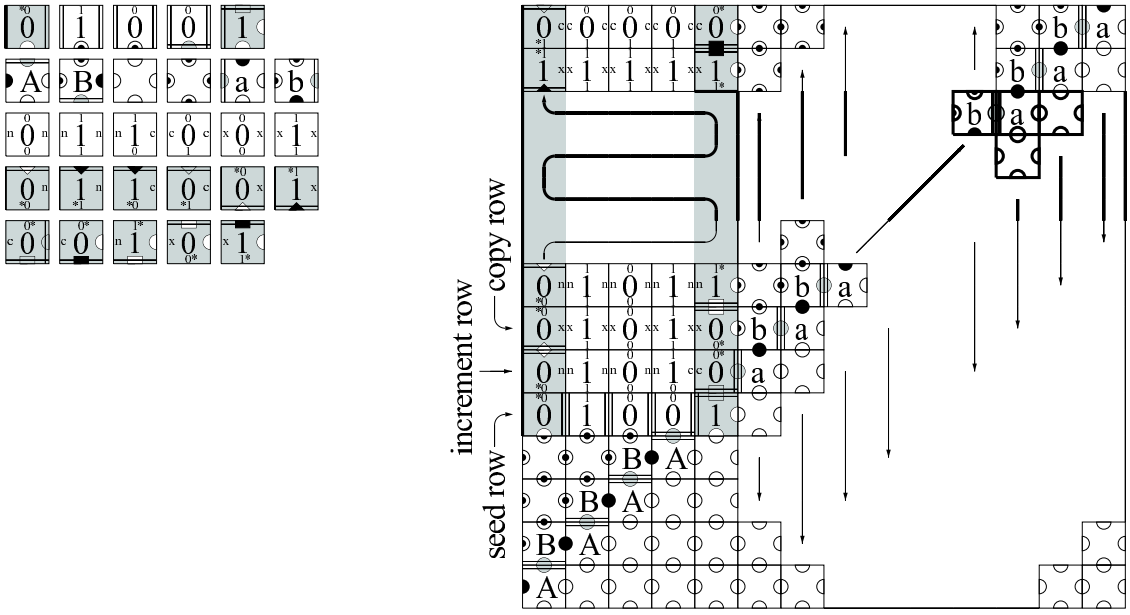


Figure 1.4: A construction for uniquely assembling an $N \times N$ square using $O(\log N)$ non-empty tile types. Thick sides have strength 0, thin sides have strength 1, and double-lined sides have strength 2. Adjacent tiles in the seed row share a unique binding domain. Source: [18].

$c - 1$ in binary. Moreover, we use special binding domains for the north side of the two tiles corresponding to the first and last digits. This is so that the self-assembly process can identify when the end of a row has been reached, and also to initiate assembly of a new row via a strength 2 bond.

From the seed row of the $(n-1) \times (n-1)$ square, self-assembly proceeds to construct a counter which counts from c to 2^{n-1} in binary, using two rows for each integer. Also, the tiles of the diagonal are assembled, which allows the rest of the square to be filled in. If $N - n$ is even, the counter starts with an *increment row*, where the number that is encoded into the binding domain of the previous row is incremented by one. If $N - n$ is odd, the counter starts with a *copy row*, which simply copies the integer that is encoded into the binding domains of the previous row. For increment rows self-assembly proceeds from left to right, whereas for copy rows it proceeds from right to left.

One can think of the counter tiles that do not correspond to first and last digits as

taking two inputs, the binding domain exposed on the neighbouring tile of the previous row, denoted p, and the binding domain exposed on the neighbouring tile of the same row, denoted s, and computing an output. If $s = x$, indicating a copy row, then the label of the tile as well as the binding domain on the north side is again p, and the binding domain on the remaining side is x (we are still in a copy row). If $s = c$, indicating a "carry" in an increment row, then the label of the tile as well as the binding domain on the north side is computed as $p + 1 \mod 2$. The binding domain on the remaining side is c, if $p = 1$ (we still need to carry), or n, if $p = 0$ (carrying completed). Similarly, if $s = n$, indicating no carrying in an increment row, the label of the tile as well as the binding domain on the north side is p. The binding domain on the remaining side is n (still no carrying).

When the leftmost digit of the counter changes from 1 to 0, presenting a binding domain on the north side thus far not encountered in the construction, a tile with a unique binding domain on its north side attaches. Since this binding domain is not found on any other tile, the self-assembly process terminates. □

Rothemund and Winfree improved this bound even further. However, while the above results apply for any value of N, the following three bounds do not hold for all values of N. To state the results, we first need some definitions.

We say proposition $P(n)$ holds *infinitely often* if and only if for every $n_0 \geq 0$, there exists $n \geq n_0$ such that $P(n)$ holds. We now define $O_{i.o.}$ ("big-O infinitely often") as follows: $f(n) = O_{i.o.}(g(n))$ if and only if there exists a constant c such that $f(n) \leq cg(n)$ infinitely often.

We say proposition $P(n)$ holds *for almost all* n if and only if

$$\lim_{x \to \infty} \frac{|\{1 \leq n \leq x \,:\, P(n)\}|}{x} = 1.$$

Now define $\Omega_{a.a}$ ("big-Ω almost always") as follows: $f(n) = \Omega_{a.a}(g(n))$ if and only if there exists a constant $c > 0$ such that $f(n) \geq cg(n)$ for almost all n.

Denote by $\log^* N$ be the least integer n such that $N \leq \underbrace{2^{2^{2^{\cdots^2}}}}_{n \text{ times}}$.

Theorem 4. [18] The tile complexity of an $N \times N$ full square at temperature 2 is $O_{i.o.}(\log^* N)$.

Sketch of Proof. The construction for this involves recursively iterating the $O(\log N)$ construction above. Starting from an $N \times N$ square constructed as in the proof of Theorem 3, the strength 2 binding domain on the north side of the tile in the north-west corner initiates a new fixed width binary counter. This counter counts from 0 to 2^N again using two rows for each integer. New tiles are introduced for this counter preventing the previous counter tiles to incorporate into the new counter. This results in an $(N + 2 \times 2^N) \times (N + 2 \times 2^N)$ full square. For the base case of $N = 2$, the $N \times N$ square from the $O(\log N)$ construction uses fewer than $22 \log^* N$ distinct non-*empty* tile types and with each iteration only 22 new tiles are introduced. Hence, the $(N + 2 \times 2^N) \times (N + 2 \times 2^N)$ full square can be assembled using at most $22(\log^* N + 1) \leq 22 \log^*(N + 2 \times 2^N)$ distinct non-*empty* tile types. □

Theorem 4 says that an infinite number of squares can be made from a relatively small number of tile types. However, Rothemund and Winfree in [18] showed that one can do even better.

Theorem 5. [18] The tile complexity of an $N \times N$ full square at temperature 2 is $O_{i.o.}(f(N))$, where $f(N)$ is any non-decreasing unbounded computable function.

While the last result shows that the number of distinct tile types required to uniquely assemble a square can be made "arbitrarily slow growing" for infinitely many squares, the next result tells us that for most squares this is not the case.

Theorem 6. [18] The tile complexity of an $N \times N$ full square at temperature 2 is $\Omega_{a.a}(\frac{\log N}{\log \log N})$.

This lower bound is matched by an upper bound of Adleman et al., who in [3] gave a construction using $O(\frac{\log N}{\log \log N})$ tile types.

Theorem 7. [3] The tile complexity of an $N \times N$ full square at temperature 2 is $O(\frac{\log N}{\log \log N})$.

Sketch of Proof. The construction again uses a binary counter, however the seed row is constructed differently. Observe that in the construction in the proof of Theorem 3 the majority of the tile types are used for the seed row, while only a constant number of tile types is used for the counter and the remainder of the square. Every tile in the seed row has

to be distinct, and as the seed encodes a binary integer, $\lceil \log N \rceil - 1$ tile types are required for the seed row alone. By encoding the number in the seed row in a larger base b instead of binary, the seed row can be shortened. However, this increases the number of tiles required for the counter. Adleman et al. optimized the number of tile types by choosing b to be a power of 2 such that

$$\frac{\log N}{\log \log N} \leq b = 2^k < \frac{2 \log N}{\log \log N}.$$

The number of tiles required for the seed row now is

$$\log_b N = \frac{\log N}{\log b} \leq \frac{\log N}{\log \log N - \log \log \log N} = O\left(\frac{\log N}{\log \log N}\right).$$

While the number of tiles required for the counter is $O(b) = O\left(\frac{\log N}{\log \log N}\right)$. □

Adleman et al. [3] also showed that their bound of $O(\frac{\log N}{\log \log N})$ can be achieved while simultaneously achieving optimal "time complexity". For this, they modified their construction: first the seed row is constructed encoding a number in base b, then a base conversion process is initiated, converting from base b to binary, and finally a binary counter is implemented where tiles can be attached in parallel. However, this construction uses temperature 3. The authors asked whether the temperature can be reduced to 2. This was answered by Cheng and Moisset de Espanes [6], who showed that the temperature can be reduced to 2 while still keeping the same asymptotic bound.

Few shapes other than squares have been considered in the literature for the standard tile assembly model. Aggarwal et al. [4] studied the tile complexity of more general rectangles. A *rectangle* R is a shape for which there exist integers $N \geq 2$ and $M \geq 2$ and a vertex (x_0, y_0) such that vertex $(x, y) \in R$ if and only if $x_0 \leq x < x_0 + N$ and $y_0 \leq y < y_0 + M$. Recall that for uniquely assembling an $N \times N$ square, a special case of a rectangle, $O(\frac{\log N}{\log \log N})$ distinct non-empty tile types suffice. Other rectangles however, can require significantly more tiles types.

Theorem 8. [4] The tile complexity of a $k \times N$ rectangle is $\Omega(\frac{N^{1/k}}{k})$.

In the same paper, Aggarwal et al. also gave an upper bound on the tile complexity of rectangles.

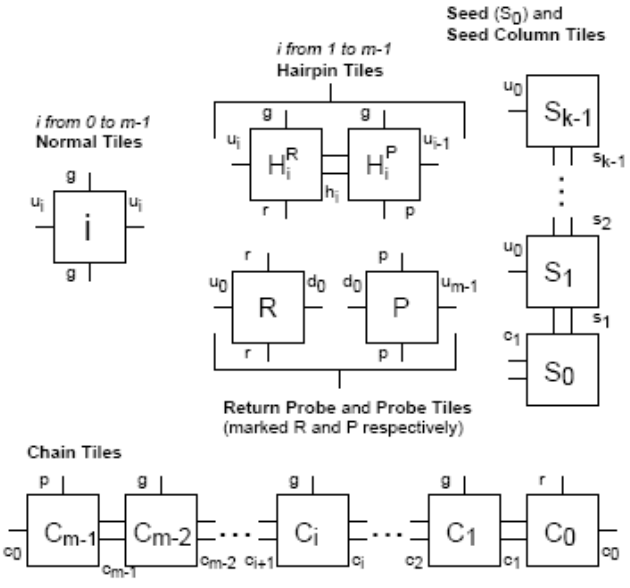


Figure 1.5: The tile set for uniquely assembling a $k \times m^k$ rectangle. Unlabeled sides represent the *null* binding domain, binding domains with a single line segment have strength 1, and binding domains with a double line segment have strength 2. Source: [4].

Theorem 9. [4] The tile complexity of a full $k \times N$ rectangle at temperature 2 is $O(N^{1/k}+k)$.

Sketch of Proof. The proof involves the construction of a k-digit base m counter, where $m = \lceil N^{1/k} \rceil$. Unlike the counter that was used in the proof of Theorem 3, each column represents a number of the counter, with the least significant digit being in the bottom row (labeled by C_0, C_1, ..., C_{m-1}). Furthermore, there are no copy rows; each row is an increment row. The basic construction, which corresponds to the tiles in Figure 1.5, assembles a $k \times m^k$ rectangle. However, by changing the binding domains encoded into the west side of the seed column tiles, the counter can be set to start at any number between 0 and $m^k - 1$. Hence, by encoding the number $m^k - N$ into the binding domains of the west side of the seed column tiles, any $k \times N$ rectangle can be assembled. □

Any shape can be produced via self-assembly by simply using a distinct tile for each vertex of the shape and a unique binding domain for every adjacent pair of tiles (recall that a shape consists of a single connected component). However, since in practice the number of distinct binding domains is a limiting factor, as each new binding domain requires significant

biochemical research and experiments, we would like to know which shapes can be assembled using a small number of tile types (and hence a small number of binding domains). Soloveichik and Winfree [22] showed that the tile complexity of a shape (where the shape is allowed to be scaled) can be bounded by the so-called Kolmogorov complexity of the shape.

To state the result we first need some definitions. Given a shape S and a positive integer c, a *c-scaling of* S is the induced subgraph of the lattice, denoted by S^c, whose vertex set is $V(S^c) = \{(i,j) \,|\, (\lfloor i/c \rfloor, \lfloor j/c \rfloor) \in V(S)\}$. Observe that this "magnification" of S by a factor of c is again a shape, i.e., it consists of a single connected component. Given two shapes S_1 and S_2, we write $S_1 \cong S_2$, if for some positive integers c and d, S_1^c can be made identical to S_2^d by translation. The relation $\cong$ is an equivalence relation. The equivalence class containing shape S is denoted by $\tilde{S}$. We say that $\tilde{S}$ is the *shape of assembly* A, if assembly A has shape S and $S \in \tilde{S}$. The *tile complexity* of $\tilde{S}$ at temperature τ, denoted by $TC^\tau(\tilde{S})$, is the minimum number of distinct non-*empty* tile types required in a tile system that uniquely produces assembly A under the standard tile assembly model at temperature τ, where $\tilde{S}$ is the shape of A. Informally, the Kolmogorov complexity of a shape S is the size of the smallest program outputting it as a list of locations. More precisely, given a universal Turing machine U, the Kolmogorov complexity of shape S, denoted by $K(S)$, is defined as $K(S) = \min\{|s| \text{ s.t. } U(s) = \langle S \rangle\}$, where $\langle S \rangle$ is an explicit binary encoding of the vertices of S. The Kolmogorov complexity of the equivalence class $\tilde{S}$, denoted by $K(\tilde{S})$, is defined as $K(\tilde{S}) = \min\{K(S) \,|\, S \in \tilde{S}\}$.

Theorem 10. [22] There exist constants a_0, a_1, b_0, b_1 such that for any equivalence class of shapes $\tilde{S}$,

$$a_0 K(\tilde{S}) + b_0 \leq TC^2(\tilde{S}) \log TC^2(\tilde{S}) \leq a_1 K(\tilde{S}) + b_1.$$

While previous result only applied for a particular shape, such as a square or rectangle, Theorem 10 applies for any shape. It is interesting to note that the construction in the proof of Theorem 10 converts each vertex of a shape S to a $c \times c$ block, where there can be binding domain mismatches between blocks. All other results in this section did not involve any binding domain mismatches.

1.3 Other Models of Self-Assembly

In an attempt to reduce tile complexity, several modifications to the standard tile assembly model have been proposed. We will mention here the *flexible glue*, *multiple temperature*, *multiple tile*, and *unique shape* models, and also look at how tile complexity can be reduced by temperature programming and staged assembly. We introduce our own modification to the standard tile assembly model, the *step assembly model*, in Chapter 3.

The Flexible Glue Model. This model differs from the standard tile assembly model only in that the restriction that the interaction strength between different binding domains is zero is removed. For this model, the lower bound of $\Omega_{a.a}(\frac{\log N}{\log \log N})$ does not apply.

Theorem 11. [4] In the flexible glue model, the tile complexity of assembling an $N \times N$ square at temperature 2 is $\Omega_{a.a}(\sqrt{\log N})$.

In [4] and [6] two constructions were given that match this lower bound.

Theorem 12. [6, 4] In the flexible glue model, the tile complexity of assembling an $N \times N$ square at temperature 2 is $O(\sqrt{\log N})$.

For both constructions, the proof is based on assembling a seed block that encodes a binary integer (using $O(\sqrt{\log N})$ tile types) and combining this with a fixed width binary counter (which uses only a constant number of tile types). Naturally, these constructions utilize binding domain mismatches.

The Unique Shape Model. In this model we redefine what it means for a tile system to uniquely produce a shape. We say a tile system $\mathbf{T}$ uniquely produces shape S, if for every assembly A of $\mathbf{T}$ there is a terminal assembly B of $\mathbf{T}$, such that $A \rightarrow_{\mathbf{T}}^{+} B$ and B has shape S. This differs from the standard tile assembly model in that the terminal assembly does not have to be uniquely produced, as in the standard tile assembly model.

The Multiple Temperature Model. To define this model, we first need some more definitions. Given a configuration C, the *adjacency graph* of C is the subgraph of the lattice

induced by the vertex set of C. A configuration is said to be *connected* if its adjacency graph is connected. Observe that an assembly is also a connected configuration. A *cut* of a connected configuration C is an edge-cut of the adjacency graph of C. Furthermore, for each edge e_i in a cut of C, define the *edge strength* of e_i to be the interaction strength (as defined by the strength function) of the binding domains on the abutting sides of the adjacent tiles at the endpoints of e_i in C. The *cut strength* of a cut is now defined as the sum of edge strengths over all edges in the cut.

In the multiple temperature model, the temperature parameter τ is replaced with a sequence of temperatures $\{\tau_i\}_{i=1}^{k}$, called the *temperature sequence* of the tile system. A tile system with k temperatures in its temperature sequence is called a *k-temperature* tile system. In such a tile system, assembly takes place in k phases. First, tiles are added to the seed tile as in the standard tile assembly model under temperature τ_1. When no more tiles can be attached, phase 2 starts. The temperature of the tile system now switches to τ_2. Now, tiles can be added as in the standard tile assembly model under τ_2, and some tiles can also break off, if their bonds are no longer strong enough under the new temperature τ_2. More specifically, if at any point during phase 2 there is a cut of the assembly with cut strength less than τ_2, then the portion of the assembly occurring on the side of the cut not containing the seed tile may be removed. Once no more tiles can be added or removed, phase 2 is complete and phase 3 starts. The temperature is set to τ_3 and tiles are now added and removed under this new temperature. This process continues until phase k is complete. An assembly A is a *terminal assembly* of a k-temperature tile system $\mathbf{T}$, if for some choice of additions and removals each of the k phases finishes and A is the resulting assembly. Furthermore, a k-temperature tile system $\mathbf{T}$ *uniquely produces* A, if the k phases always finish regardless of the choice of additions and removals, and A is the unique terminal assembly of $\mathbf{T}$.

The Multiple Tile Model. In this model, tiles can bind to each other before being attached to the growing assembly. More specifically, a tile set T and a temperature τ have a corresponding set of *addable supertiles* $W(T, \tau)$. A *supertile* is a finite, connected configuration. Every tile $t \in T$ has an associated supertile, namely C_t (recall that C_t is the configuration that maps $(0, 0)$ to t and every other vertex is mapped to the *empty* tile). The set $W(T, \tau)$ is defined recursively: (i) $C_t \subseteq W(T, \tau)$ for every tile $t \in T$, (ii) for any

$A, B \in W(T, \tau)$, if A and B can be abutted together (without overlap) to form another supertile $A \oplus B$, such that the total interaction strength of all abutting edges of A and B is at least τ, then $A \oplus B$ is also in $W(T, \tau)$. Self-assembly in the multiple tile model takes place as in the standard tile assembly model, with the exception that instead of adding tiles from T to the growing complex, we now add supertiles from $W(T, \tau)$.

Note that since the standard tile assembly model is a special case of each of the multiple temperature model, and the unique shape model, the upper bound on the tile complexity of $N \times N$ squares of $O(\frac{\log N}{\log \log N})$ (from Theorem 7) still applies for each of these new models.

Aggarwal et al. [4] extended the lower bound of Theorem 6 to apply to other models of self-assembly.

Theorem 13. [4] The tile complexity of an $N \times N$ square is $\Omega_{a.a}(\frac{\log N}{\log \log N})$ for the multiple tile model, the multiple temperature model (with constant size temperature sequence), and the unique shape model.

Hence, for squares, the multiple tile, multiple temperature, and unique shape model do not offer an improvement over the standard tile assembly model.

Aggarwal et al. [4] also considered more general rectangles other than squares under various models of self-assembly.

Theorem 14. [4] The tile complexity of a $k \times N$ rectangle is $O(N^{1/k} + k)$ for the flexible glue model, the multiple tile model, and the unique shape model.

Again, this is the same bound as for the standard tile assembly model. However, the bound can be improved when considering the multiple temperature model. The idea is to first assemble a larger rectangle under τ_1 with a suitable cut of cut strength less than τ_2. Once the temperature is raised to τ_2 part of the shape falls off and the desired rectangle is produced.

Theorem 15. [4] The tile complexity of a $k \times N$ rectangle, where $k \geq 2$, is $O(\frac{\log N}{\log \log N})$ for the two-temperature model.

Aggarwal et al. [4] also gave a lower bound on the tile complexity of rectangles in the new models of self-assembly. This lower bound is an extension of Theorem 8.

Theorem 16. [4] The tile complexity of a $k \times N$ rectangle is $\Omega(\frac{N^{1/k}}{k})$ for the flexible glue model and the unique shape model.

In light of these results, Aggarwal et al. [4] posed several questions. Can new examples be found showing that the new models reduce tile complexity? Are there shapes for which the unique shape model reduces tile complexity (recall that the unique shape model did not reduce the tile complexity of squares or thin rectangles compared to the standard model)? Also, for the multiple temperature model, can tile complexity results be improved by using more than two temperatures? If so, is it enough do monotonically increase the temperature of the system or does it help to raise and lower the temperature? Furthermore, for the standard tile assembly model, the lower bound of Theorem 6 does not apply if the temperature of the system is a large exponential function of N. In this case, is it possible to reduce the tile complexity of $N \times N$ squares?

Two of these questions have been answered by Kao and Schweller [10], namely whether tile complexity can be reduced by using more than two temperatures and whether it can help to raise and lower the temperature of the system. Kao and Schweller considered reducing tile complexity through "temperature programming". This differs from the multiple temperature model in that temperature sequences of non-constant size are allowed. Kao and Schweller provided a construction for assembling an $N \times N$ square in the multiple temperature model using $O(1)$ tile types and a temperature sequence of length $O(\log N)$ which uses temperatures 4, 9, 3, and 7. The resulting square is not full and does have some binding domain mismatches.

Theorem 17. [10] The tile complexity of an $N \times N$ square is $O(1)$ under the multiple temperature model.

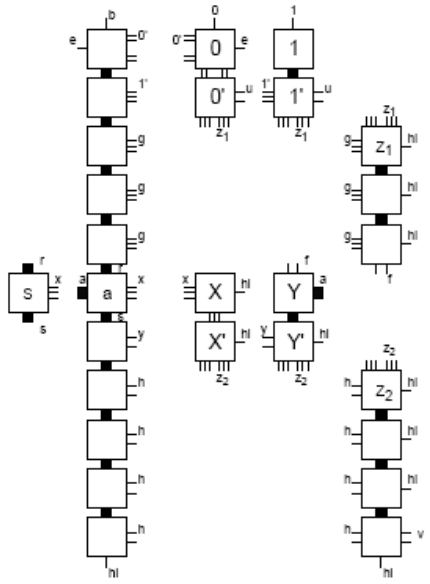


Figure 1.6: The tile set for the seed block. The number of line segments on the side of a tile represents the strength of the corresponding binding domain. Thick lines indicate binding domains of strength 9. Source: [10].

Sketch of Proof. The construction again uses a binary counter. Using a $O(\log N)$ temperature sequence the seed block of the counter can be assembled using $O(1)$ tile types. Figure 1.6 shows the tile set used for assembling the seed block. The specific temperature sequence used depends on the integer to be encoded into the seed block. Figure 1.7 illustrates the construction of a seed block that encodes the number 1010010. Self-assembly starts at temperature 4 which ensures that a digit of 0 is encoded into the assembly. Whenever the most recent 0 that was encoded should be replaced with a 1, the temperature of the tile system is raised to 9. This causes the 0-tile and its neighbour to the south to fall off, as they no longer bind strongly enough, and the 1-tile is encoded into the assembly. Next the temperature is dropped to 3, which causes the bottom half of the last column of the assembly to fill in. The tile X is used to control the growth of the assembly (it indicates that the least significant digit of the string to be encoded has been reached). If self-assembly is to continue, the temperature of the system is raised to 7. This removes the tile X and its neighbour to the south and attaches tiles Y and a. Now the temperature is again set to 4 and the above process is repeated to encode another digit. In general this results in an $11 \times 2m$ seed block, where m is the length of the encoded binary string. Once the seed

block is completed, the temperature is dropped to 2. This allows the binary counter tiles (and the tiles for the diagonal as well as the "blank" tiles above and below the diagonal) to attach and complete the assembly into a square. □

Kao and Schweller showed that their result is optimal, in the sense that for any tile set T that uniquely assembles an $N \times N$ square under temperature sequence $\{\tau_i\}_{i=1}^{k}$, for almost all N it cannot be the case that both $|T| = o(\frac{\log N}{\log N \log N})$ and $k = o(\log N)$.

Kao and Schweller also provided a modified construction that is robust against certain types of assembly errors. Note that the tile system in the proof of Theorem 17 does not uniquely produce the desired square, if tiles bind to each other before being attached to the growing assembly (as in the multiple tile model). For example, if in the temperature 4 phase supertiles are added and removed as in the multiple tile model, then the z_1-tile can bind with its neighbour to the south with their strength 9 binding domain. Together they can now attach to the assembly before the 0-tile is added. This causes both the 0-tile and the 1-tile to be able to attach and the ability of the tile system to encode a specified binary integer is lost. Kao and Schweller showed that even if tiles bind to each other before being attached to the growing assembly, the bound of Theorem 17 still holds.

Theorem 18. [10] The tile complexity of an $N \times N$ square is $O(1)$ under the multiple temperature model, where in each phase tiles are added and removed under the multiple tile model.

These are the first constructions that assemble $N \times N$ squares using a constant number of tile types. Kao and Schweller asked whether it is possible to have a general shape building tile set that can be programmed to assemble into any arbitrary shape via a temperature sequence that encodes a description of the shape. They also asked if the tile complexity and the length of the temperature sequence can be reduced by increasing the value of the temperatures in the temperature sequence.

Demaine et al. [7] provided other constructions for assembling squares using a constant number of tile types. They also gave constructions for assembling more general shapes using

$O(1)$ tile types. Their results are based on the staged assembly model.

The Staged Assembly Model. This model differs from the standard tile assembly model in that tiles can be added in stages (rather than all tiles being present at the start of the assembly process) and non-attached tiles can be filtered out after each stage. Also, different assemblies can be built in different "bins" (using different tile sets or supertiles) and then the contents of these bins can be mixed allowing the assemblies to be combined to form larger assemblies. More precisely, assembly starts with any number of bins. Each bin contains single tiles that, during the first stage, self-assemble as in the multiple tile assembly model. For each bin, only the terminal assemblies are retained and the remaining tiles are filtered out. In subsequent stages, any collection of operations of the following two types can be performed: (i) add arbitrarily many copies of a new tile to an existing bin, and (ii) pour the contents of some bin A into another bin B, mixing the contents of bin A into bin B and keeping bin A intact. In each stage now, self-assembly proceeds as in the multiple tile model, as we are no longer attaching only single tiles, but also supertiles. All bins at any stage are assumed to have the same temperature τ. After each stage, only "terminally produced supertiles" are retained in their bins and other supertiles are filtered out. Intuitively, a supertile is terminally produced, if no supertiles can be attached to it. Furthermore, at every stage it is assumed that the terminally produced supertiles are uniquely produced, that is, every supertile can be grown into a terminally produced supertile. Every staged tile system has an associated "mix graph" which specifies precisely how to mix the bins when transitioning from one stage to the next. When only a single bin and a single stage are used, one obtains the standard tile assembly model. Demaine at al. focused on the "glue complexity" (the number of non-*null* binding domains in the tile system) rather than the tile complexity (the number of non-*empty* tile types). In addition, they also considered the "stage complexity" (the number of stages) and the "bin complexity" (the number of bins).

As in the temperature programming model, also in the staged assembly model, squares can be assembled using a constant number of tile types. However, while the construction for the temperature programming model yielded squares with binding domain mismatches, the following theorem applies for full squares.

Theorem 19. [7] The tile complexity of an $N \times N$ full square is $O(1)$ under the staged

assembly model with temperature 1.

Demaine et al. [7] showed that also more general shapes can be assembled using a small number of tile types.

Theorem 20. [7] Any shape S can be uniquely assembled using at most 2 non-*null* binding domains, and at most 52 non-*empty* tile types under the staged assembly model with temperature 1.

While Theorem 20 produces non-full shapes, Demaine et al. [7] showed that also full shapes which are hole free can be produced using a constant number of tile types when scaling by a factor of 2 is allowed.

Theorem 21. [7] Any simply connected full shape S can be uniquely assembled using a scale factor of 2, 8 non-*null* binding domains, and $O(1)$ tile types, under the staged assembly model with temperature 1.

Based on their work, Demaine et al. [7] suggested several further research directions. Can the assumption that all supertiles assemble to completion be relaxed? It is plausible that in practice some supertiles may not reach their terminal state. Can staged assembly systems detect such errors? Another direction would be to extend the model to 3 dimensions. In practice, 3D assembly is much harder than 2D assembly. This is partly due to the fact, that in practice 2D assembly makes use of 3 dimensions. Consider for example two supertiles in the plane with complex borders. It may not be possible to "slide" the two supertiles together within that plane; one supertile might have to be "lifted up" and dropped into position. A further direction is to consider nondeterministic assembly, where tile systems can build a large class of shapes, rather than uniquely produce a shape. Can a tile system be designed such that certain shapes are assembled with high probability?

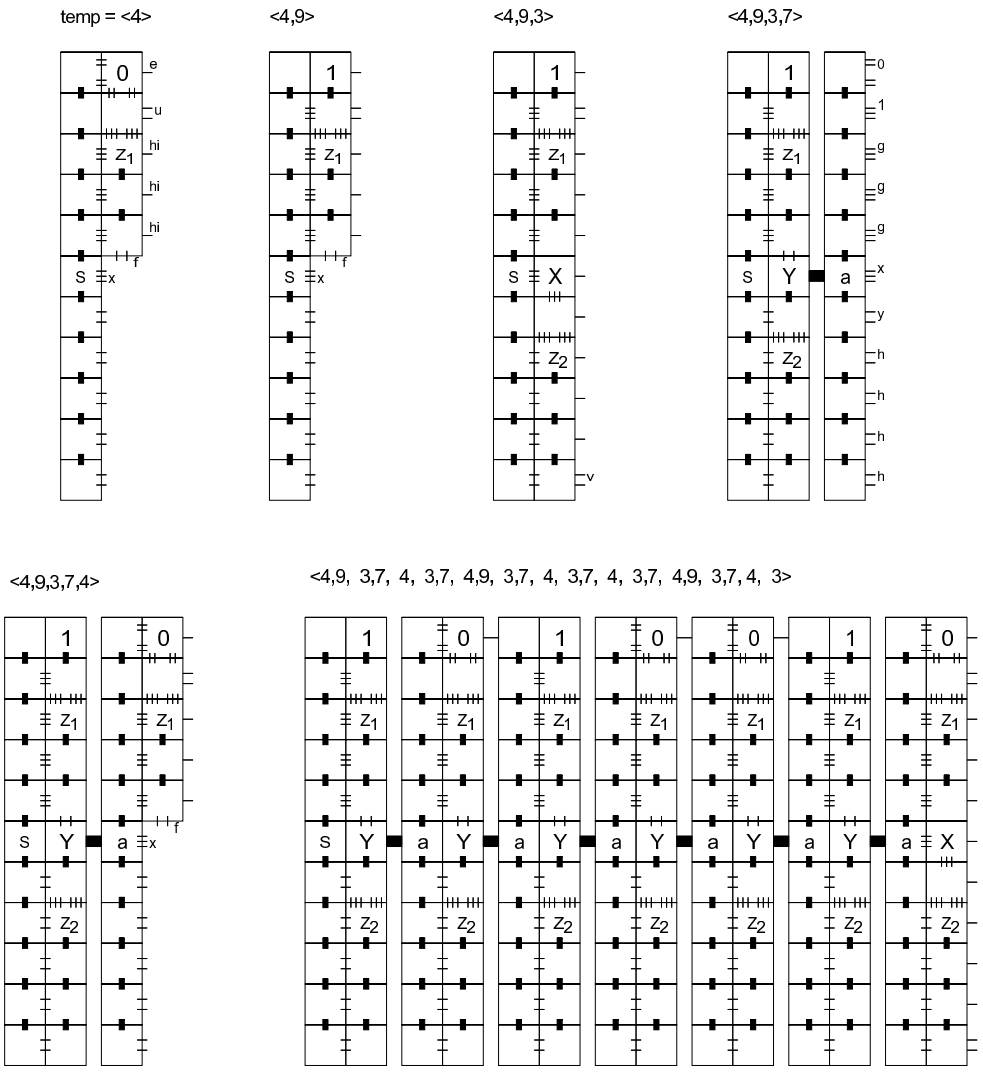


Figure 1.7: The construction of a seed block encoding 1010010. Source: [10].

Chapter 2

Lower Bounds for Assemblies at Temperature 1

In this chapter we give two lower bounds on the minimum number of tile types needed to uniquely assemble an arbitrary shape in the standard tile assembly model at temperature 1. The first result is based on the radius of the shape, and the second result is based on the Manhattan diameter of the shape. Applying the second result to a square provides a partial answer to an open problem of Rothemund and Winfree. Recall that Rothemund and Winfree conjectured that $2N - 1$ distinct tile types are required to uniquely assemble an $N \times N$ square under the standard tile assembly model at temperature 1. Our result implies that $2N - 1$ distinct tile types are indeed required, if we assume that the square uniquely assembles without any binding domain mismatches. Most results in this section only hold under the assumption of no binding domain mismatches, i.e., any two adjacent non-*empty* tiles either form a bond, or the abutting sides of the tiles both have the binding domain *null* assigned to them. Without this assumption the tile system would be prone to errors during a self-assembly process. For instance, consider the assembled shape in Figure 2.2(b). This assembly contains one binding domain mismatch: between the seed tile S and tile 5. Even though this assembly is terminal (cannot be extended), there is a chance that another copy of tile 6 would push out and replace the seed tile which would lead to incorrectly assembling infinitely growing assemblies. We also note that most constructions used in the literature are without binding domain mismatches.

All tile systems in this chapter are assumed to be standard tile systems. Furthermore all tile systems use the strength function s_Σ which satisfies $s_\Sigma(\sigma, \sigma) = 1$ for all $\sigma \in \Sigma \backslash \{empty\}$.

2.1 Lower Bound based on Graph Radius

The following Lemma is based on the argument in the proof of Theorem 1 showing that all tiles in a full square at temperature 1 need to be distinct. While here we do not require the assembly to be full, we do insist that there are no binding domain mismatches. Recall that an assembly A is full, if for any two neighbouring positions (x, y) and $d(x, y)$ in $V(A)$, where $d \in \mathcal{D}$, there is a bond between them in A.

Lemma 1. *Let $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ be a tile system that uniquely produces a finite assembly A. Suppose there are no binding domain mismatches in A. Let P be a path in the backbone graph of A with one endpoint at $(0, 0)$ (the position of the seed tile). Then all tiles of A on P are distinct.*

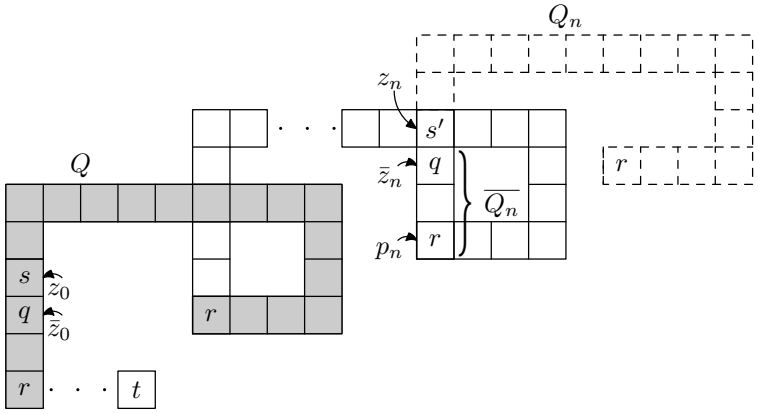


Figure 2.1: Forming $(A[R] \cup B_{n-1}) \cup A_n[Q_n]$ with a conflict at vertex z_n.

Proof. We will think of P as being directed, with $(0, 0)$ being the first vertex of P. Towards a contradiction, suppose a tile r is repeated twice on P. Say r occurs at the two positions $p_0 = (x_0, y_0)$ and $p_1 = (x_1, y_1)$, where p_0 precedes p_1 on P. Let Q be the subpath of P from p_0 to p_1 (including p_0 and p_1). Let ϕ be the translation mapping p_0 to p_1. For every integer $n \geq 0$, let $Q_n = \phi^{(n)}(Q)$ be the translated version of the path Q, i.e., the vertex (x, y) of Q corresponds to the vertex $\phi^{(n)}(x, y)$ of Q_n. Similarly, for every $n \geq 0$, let $A_n = \phi^{(n)}(A)$

be the translated version of configuration A so that $A_n(\phi^n(x,y)) = A(x,y)$. Let R be the subpath of P from $(0,0)$ to p_0. Recall that $A[R]$ is the subconfiguration of A induced by the vertices of R.

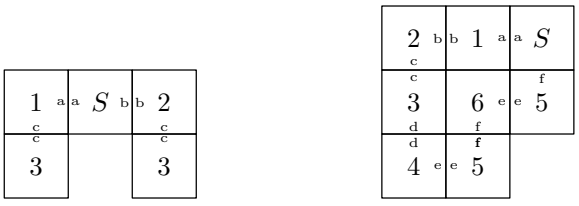
As in the proof of Theorem 1, we will show that, starting from the assembly corresponding to R, we can "add" translated versions of Q infinitely often. More precisely, let $B_n = \bigcup_{i=0}^{n} A_i[Q_i]$. We will show that $A[R] \cup B_n \in Asmb(\mathbf{T})$ for all $n \geq 0$. This leads to a contradiction, since $\mathbf{T}$ uniquely produces a finite assembly. To prove the claim, it suffices to show that for all $n \geq 0$, $A[R] \cup B_n$ is a configuration, its vertex set contains $(0,0)$ (since every assembly starts from the seed tile), and its backbone graph is connected (since tiles can only attach to the growing complex if they form a bond). Note that since $(0,0) \in V(R)$, $(0,0)$ is in the vertex set of $A[R] \cup B_n$ for every n. We will prove the remaining two conditions by induction on n.

Fist suppose $n = 0$. Observe that $B_0 = A_0[Q_0] = A[Q]$. Since p_0 is the only vertex in $V(R) \cap V(Q)$, and $A[R](p_0) = A[Q](p_0)$, $A[R] \cup A[Q]$ is again a configuration of T. The backbone graphs of $A[R]$ and $A[Q]$ are connected and share a vertex, hence, also the backbone graph of $A[R] \cup A[Q]$ is connected. Thus, $A[R] \cup A[Q] \in Asmb(\mathbf{T})$.

Now suppose the claim also holds for some $n - 1 \geq 0$. Towards a contradiction, suppose $A[R] \cup B_n = (A[R] \cup B_{n-1}) \cup A_n[Q_n]$ is not a configuration. Then there must exist a vertex (x,y) on the path Q_n such that $(A[R] \cup B_{n-1})(x,y) = s' \neq s = A_n[Q_n](x,y)$, where s', $s \neq empty$. Let z_n be the first such vertex on Q_n starting from $p_n = \phi^{(n)}(p_0)$ (see Figure 2.1). Observe that $z_n \neq p_n$, since $(A[R] \cup B_{n-1})(p_n) = A_n[Q_n](p_n) = r$. Let $\bar{z}_n$ be the neighbour of z_n on Q_n that is closer to p_n, say $\bar{z}_n = d(z_n)$ for some direction $d \in \mathcal{D}$ ($d = S$ in Figure 2.1). Let $\bar{Q}_n$ be the subpath of Q_n from p_n to $\bar{z}_n$. By our choice of $\bar{z}_n$, $(A[R] \cup B_{n-1}) \cup A_n[\bar{Q}_n]$ is a configuration of T. By the induction hypothesis, the backbone graph of $A[R] \cup B_{n-1}$ is connected. Since the backbone graph of $A_n[\bar{Q}_n]$ is isomorphic to a subpath of P, it is connected. Furthermore, since the vertex sets of $A[R] \cup B_{n-1}$ and $A_n[\bar{Q}_n]$ intersect in p_n, the backbone graph of $(A[R] \cup B_{n-1}) \cup A_n[\bar{Q}_n]$ is connected. Thus, $(A[R] \cup B_{n-1}) \cup A_n[\bar{Q}_n] \in Asmb(\mathbf{T})$. This implies that $(A[R] \cup B_{n-1}) \cup A_n[\bar{Q}_n]$ is a subconfiguration of the unique terminal assembly A.

Let q be the tile at position $\bar{z}_n$ in $A_n[Q_n]$. The tile q also appears at position $\bar{z}_0 = \phi^{(-n)}(\bar{z}_n)$ in $A[Q]$. Since in $A[Q]$ there is a bond between the adjacent positions $\bar{z}_0$ and z_0, the binding domain of tile q on side d^{-1} cannot be *null*. This implies that in $(A[R] \cup B_{n-1}) \cup A_n[\bar{Q}_n]$ there must also be a bond between the adjacent positions $\bar{z}_n$ and z_n, as $(A[R] \cup B_{n-1}) \cup A_n[\bar{Q}_n]$ is a sub-configuration of the unique terminal assembly A which does not have any binding domain mismatches. Thus, both the tile s and the tile s' can bind to q on side d^{-1}, i.e., $s_d = s'_d = q_{d^{-1}}$. Now let $\bar{Q} = \phi^{(-n)}(\bar{Q}_n)$, and let C be the configuration defined by $C(z_0) = s'$ and $C(x, y) = empty$ for $(x, y) \neq z_0$. Observe that $A[R] \cup A[\bar{Q}] \in Asmb(\mathbf{T})$. Since s' can bind to q on side d^{-1}, and $(A[R] \cup A[\bar{Q}])(z_0) = empty$, it follows that $(A[R] \cup A[\bar{Q}]) \cup C \in Asmb(\mathbf{T})$. This contradicts the uniqueness of assembly A, as A and $(A[R] \cup A[\bar{Q}] \cup C)$ differ at vertex z_0. Therefore, $A[R] \cup B_n = (A[R] \cup B_{n-1}) \cup A_n[Q_n]$ must be a configuration of T. By the induction hypothesis, the backbone graph of $A[R] \cup B_{n-1}$ is connected. Since the backbone graph of $A_n[Q_n]$ is a translated version of path Q, it is connected. Furthermore, since the vertex sets of $A[R] \cup B_{n-1}$ and $A_n[Q_n]$ intersect in p_n, also the backbone graph of $(A[R] \cup B_{n-1}) \cup A_n[Q_n]$ is connected. Hence $(A[R] \cup B_{n-1}) \cup A_n[Q_n] = A[R] \cup B_n \in Asmb(\mathbf{T})$.

Therefore, $A[R] \cup B_n \in Asmb(\mathbf{T})$ for all $n \geq 0$. Since every assembly of $\mathbf{T}$ is a subconfiguration of the unique terminal assembly A, $A[R] \cup B_n$ is a subconfiguration of A for every $n \geq 0$. As A is finite, this is a contradiction. □



(a) (b)

Figure 2.2: Fewer tiles suffice if we do not require the seed tile to be an endpoint of the path (a), or do not require that there are no binding domain mismatches (b).

Note that the conclusion of the statement of Lemma 1 is false, if we drop either the condition that the seed tile is an endpoint of the path, or the condition that there are no binding domain mismatches. Figure 2.2(a) shows that if we do not require the position of the seed tile to be an endpoint of the path (while still keeping the assumption of no binding

domain mismatches), we can uniquely assemble a path consisting of five vertices using only four distinct tile types. Figure 2.2(b) shows that if we allow binding domain mismatches we can uniquely assemble a path starting at the position of the seed tile and consisting of eight vertices using only seven distinct tile types (note the mismatch between tile S and tile 5).

Lemma 1 says that in assemblies at temperature 1 all tiles on a path from the seed tile need to be distinct (under the assumption of no binding domain mismatches). However, there may still be tiles that are repeated in the assembly. We will use Lemma 1 to show that assemblies at temperature 1 whose backbone graph is 2-connected require *all* of their tiles to be distinct.

Corollary 1. *Let* $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ *be a tile system that uniquely produces a finite assembly* A*. If* A *has no binding domain mismatches and the backbone graph of* A *is 2-connected, then all tiles of* A *are distinct, i.e.,* $\mathbf{T}$ *contains at least* $|V(A)|$ *tile types. In particular, if* A *is a full assembly and its backbone graph is 2-connected then all tiles of* A *are distinct.*

Proof. Towards a contradiction suppose that tile r is repeated in A. Say r occurs at the two positions $p_0 = (x_0, y_0)$ and $p_1 = (x_1, y_1)$. Let G be the backbone graph of A. Since G is 2-connected, there exist two vertex disjoint paths P_1 and P_2 from p_0 to p_1. Without loss of generality let P_1 be the path such that a shortest path F from $(0, 0)$ to P_1 does not contain an internal vertex of P_2. Then the path consisting of F, the segment of P_1 from the intersection with F to an endpoint of P_1, and P_2, is a path starting at $(0, 0)$ on which tile r appears twice. This contradicts Lemma 1. Hence, all tiles of A are distinct. □

While Lemma 1 only applies to paths starting from the position of the seed tile, it also limits how many times a tile can be repeated on an arbitrary path in the backbone graph. In particular, on *any* path in the backbone graph a tile can be repeated at most twice.

Corollary 2. *Let* $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ *be a tile system that uniquely produces a finite assembly* A*. Suppose there are no binding domain mismatches in* A*. Let* P *be a path in the backbone graph of* A*. If a tile is repeated on* P*, it is repeated exactly twice on* P*. Moreover,*

if a tile repeats at vertices p_0 and p_1 on P, then the first intersection of any path in the backbone graph starting at vertex $(0,0)$ with P lies strictly between p_0 and p_1.

Proof. Towards a contradiction suppose tile r is repeated three times on P. Let u and v be the endpoints of P. Let F be a shortest path from $(0,0)$ to P, and let w be the endpoint of F that is also on P. Then, by the pigeonhole principle, either the sub-path of P from w to u or the subpath of P from w to v has tile r repeated twice. Thus, the path consisting of F together with this subpath of P, is a path starting at $(0,0)$ on which tile r appears twice. This contradicts Lemma 1. Hence, if a tile is repeated on P, it must be repeated exactly twice. The proof for the second part is similar. □

Corollary 3. *Let $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ be a tile system that uniquely produces a finite assembly A. Suppose there are no binding domain mismatches in A. Let C be a cycle in the backbone graph of A. Then all tiles of A on C are distinct.*

Proof. The argument is similar to the proof of the previous corollary. Suppose there is a tile r that appears twice on C. Say r appears at position p_0 and p_1. Since the backbone graph of A is connected, there is a path starting at $(0,0)$ that contains both p_0 and p_1. This contradicts Lemma 1. □

The *eccentricity* of a vertex v of a connected graph G is defined as $\max_{u \in V(G)} d(u,v)$, where $d(u,v)$ is the length of a shortest path from u to v. The *radius of G* is the minimum eccentricity of the vertices of G. The *radius of an assembly A*, denoted by $\mathrm{rad}(A)$, is the radius of the sub-graph of the lattice induced by V(A).

Theorem 22. *Let $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ be a tile system that uniquely produces a finite assembly A. Suppose there are no binding domain mismatches in A. Then T contains at least* $\mathrm{rad}(A) + 1$ *non-empty tiles.*

Proof. By the definition of $\mathrm{rad}(A)$, there is a path P in the backbone graph of A starting at $(0,0)$ whose length is at least $\mathrm{rad}(A)$. By Lemma 1 all tiles on P are distinct. Since there are at least $\mathrm{rad}(A) + 1$ vertices in P, there must be at least $\mathrm{rad}(A) + 1$ distinct non-empty tile types in T. □

2.2 Lower Bound based on Manhattan Diameter

Let P be a path in the backbone graph of an assembly A and assign an orientation to P. For any vertex v on P, we denote the side on which P enters v by $in^P(v)$ and the side on which P leaves v by $out^P(v)$. Let r be a tile that is repeated (exactly twice) on P, say r occurs at positions p_0 and p_1, where p_0 precedes p_1 on P. We say r has a *good repetition* if $out^P(p_0) = in^P(p_1)$. Otherwise, r has a *bad repetition*. Note that the definition of good repetition is independent of the orientation assigned to P. Also note that the repetition of tile 5 in Figure 2.2(b) is a bad repetition.

Given two vertices (x, y) and (x', y') in $\mathbb{Z} \times \mathbb{Z}$, their *Manhattan distance* is defined as $|x - x'| + |y - y'|$. The *Manhattan diameter* of an assembly A, denoted by $\text{Mdiam}(A)$, is the maximum Manhattan distance between any two vertices in $V(A)$.

We prove our main result of this section in two steps. First we show that our bound holds for paths that do not contain any bad repetitions. In the second step we show that between any two vertices in the backbone graph we can always find such a path.

Lemma 2. *Let* $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ *be a tile system that uniquely produces a finite assembly* A*. Let* P *be a path in the backbone graph of* A *such that the Manhattan distance between the endpoints of* P *is* $\text{Mdiam}(A)$*. If* P *does not contain any bad repetitions, then* T *contains at least* $\text{Mdiam}(A) + 1$ *non-empty tiles.*

Proof. Assign an orientation to P and let a be the first and b be the last vertex of P. Consider a direction $d \in \mathcal{D}$. If the Manhattan distance between $d(a)$ and b is less than the Manhattan distance between a and b, we say that d is a *forward direction.* If neither d nor d^{-1} satisfy this definition, we pick one of them to be a forward direction. We call an edge uv of P (with tail u and head v) a *forward edge* if $v = d(u)$ and d is a forward direction. Otherwise, the edge uv is a *backward edge*. Let n_1 be the number of tiles occurring exactly once on P, and let n_2 be the number of tiles that are repeated on P.

We will first show that any tile can be repeated at most twice on P. Suppose, for a contradiction, that a tile is repeated three times on P. Say it occurs at positions p_0, p_1, and

p_2 (in this order). Since P does not contain any bad repetitions, $out^P(p_0) = in^P(p_1)$ (from the good repetition p_0, p_1) and $out^P(p_1) = in^P(p_2)$ (from the good repetition p_1, p_2). This implies that $out^P(p_0) = in^P(p_1) \neq out^P(p_1) = in^P(p_2)$, making the repetition p_0, p_2 a bad repetition. Clearly, this contradicts our assumption.

To complete the proof we will use a counting argument. We will first count pairs (e, v), where v is a vertex of P that is incident with backward edge e of P. Let e_b be the number of backward edges in P. Since every backward edge is incident with exactly two vertices, there are $2e_b$ such pairs. On the other hand, for any repeated tile r occurring at positions p_0 and p_1 (where p_0 precedes p_1 on P), $out^P(p_0) = in^P(p_1)$. This implies that there is at least one backward edge incident with either p_0 or p_1. Hence, the number of (e, v) pairs is at least n_2. Combining our two counts, we see that $e_b \geq n_2/2$. By the definition of forward and backward edges, there are $\text{Mdiam}(A)$ more forward edges than backward edges in P. Thus, there are at least $\text{Mdiam}(A) + n_2/2$ forward edges in P, and hence, at least $\text{Mdiam}(A) + n_2$ edges in P in total. On the other hand, the number of edges in P is one less than the number of vertices in P. Since every tile on P appears either exactly once or exactly twice on P, the number of vertices of P is $n_1 + 2n_2$. Putting these bounds together, we see that the number of edges of P is $|P| - 1 = n_1 + 2n_2 - 1 \geq \text{Mdiam}(A) + n_2$. This implies that $n_1 + n_2 \geq \text{Mdiam}(A) + 1$. Since the set of tiles occurring exactly once on P and the set of tiles being repeated on P are disjoint subsets of T, T contains at least $n_1 + n_2 \geq \text{Mdiam}(A) + 1$ distinct non-empty tiles. □

Note that Lemma 2 holds without the assumption of no binding domain mismatches. To complete the proof of the main result we need one more Lemma.

Lemma 3. *Let $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ be a tile system that uniquely produces a finite assembly A. Suppose there are no binding domain mismatches in A. Let u and v be two vertices in $V(A)$. Then there is a path P in the backbone graph of A connecting u and v without any bad repetitions.*

Proof. Among all paths between u and v in the backbone graph of A, let P be a path with the fewest number of bad repetitions. Again, we will think of P as being directed, with u being the first vertex of P. If P does not have any bad repetitions, we are done. Hence,

assume some tile r on P has a bad repetition. Say r occurs at positions p_0 and p_1, where p_0 precedes p_1 on P. Observe that by Corollary 2, r cannot appear anywhere else on P. We will show that there exists another path from u to v having fewer bad repetitions than P, contradicting the minimality of P.

To this end, let F be a path in the backbone graph of A from $(0,0)$ to P. Let w be an endpoint of F that is also on P. By Corollary 2, w is strictly between p_0 and p_1. Let ϕ be the translation mapping p_0 to p_1, and let Q be the subpath of P from p_0 to p_1 (including both p_0 and p_1). As in the proof of Lemma 1 let $Q_1 = \phi(Q)$ be the translated version of the path Q, i.e., the vertex (x, y) of Q corresponds to the vertex $\phi(x, y)$ of Q_1. Similarly, let $A_1 = \phi(A)$ be the translated version of the configuration A, such that $A_1(\phi(x, y)) = A(x, y)$. Let R be the subtree of the backbone graph consisting of F and Q. Since $(0,0) \in V(R)$, $A[R]$ is an assembly of $\mathbf{T}$.

Similarly to the proof of Lemma 1 we will first show that $V(R)$ and $V(Q_1)$ intersect in a vertex $z_1 \neq p_1$, and that the tile in $A[R]$ at position z_1 makes a bond with its neighbouring tile on Q.

Suppose $A[R] \cup A_1[Q_1]$ is a configuration of T, then it must also be an assembly of T, since R and Q_1 share a vertex. If the backbone graph of $A[R] \cup A_1[Q_1]$ is a tree, then it contains a path starting at $(0,0)$ on which tile r appears twice (at p_1 and $p_2 = \phi(p_1)$). This contradicts Lemma 1. Thus, either the backbone graph of the assembly $A[R] \cup A_1[Q_1]$ is not a tree, or $A[R] \cup A_1[Q_1]$ is not a configuration. In either case there exists a vertex $(x, y) \neq p_1$ such that $A[R](x, y) \neq \mathit{empty}$ and $A_1[Q_1] \neq \mathit{empty}$. Let z_1 be the first such vertex on Q_1 starting from p_1. Let $\bar{z}_1$ be the neighbour of z_1 on Q_1 closer to p_1, say $\bar{z}_1 = d(z_1)$ for some direction $d \in \mathcal{D}$. Let $\bar{Q}_1$ be the subpath of Q_n from p_1 to $\bar{z}_1$. By our choice of $\bar{z}_1$, $A[R] \cup A_1[\bar{Q}_1]$ is a configuration of T. Furthermore, since $A[R]$ is an assembly of $\mathbf{T}$, and the vertex sets of $A[R]$ and $A_1[\bar{Q}_1]$ intersect, also $A[R] \cup A_1[\bar{Q}_1]$ is an assembly of $\mathbf{T}$. Hence, $A[R] \cup A_1[\bar{Q}_1]$ is a subconfiguration of A, since A is the unique terminal assembly of $\mathbf{T}$.

Let q be the tile at position $\bar{z}_1$ in $A_1[Q_1]$. The tile q also appears at position $\bar{z}_0 = \phi^{-1}(\bar{z}_1)$ in $A[Q]$. Since in $A[Q]$ there is a bond between the adjacent positions $\bar{z}_0$ and $z_0 = \phi^{-1}(z_1)$, the binding domain of tile q on side d^{-1} cannot be *null*. This implies that in $A[R] \cup A_1[\bar{Q}_1]$

there must also be a bond between the adjacent positions $\bar{z}_1$ and z_1, as $A[R] \cup A_1[\bar{Q}_1]$ is a sub-configuration of the unique terminal assembly A which does not have any binding domain mismatches.

Depending on the position of z_1 we distinguish two cases:

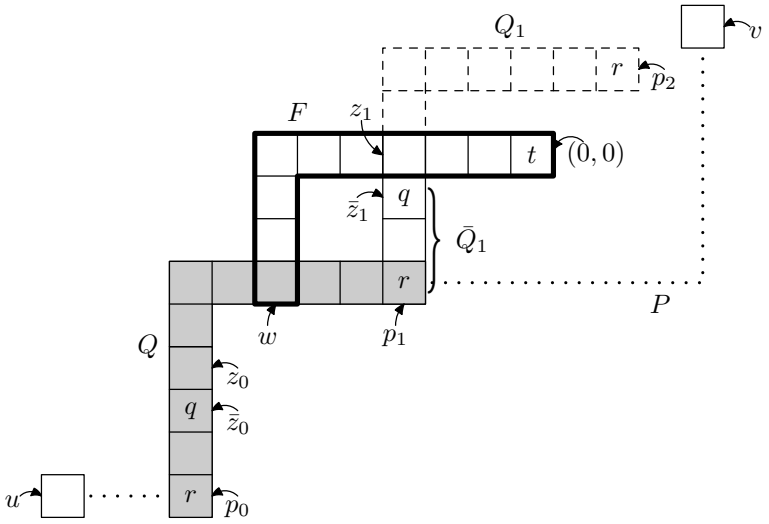


Figure 2.3: Intersection vertex z_1 is on the path F. Then there exists a path with one endpoint at $(0,0)$ on which tile r appears twice.

Case 1: z_1 is on F and $z_1 \neq w$ (see Figure 2.3).

Then there is a path starting at $(0,0)$ on which tile r appears twice; namely, the path consisting of the subpath of F from $(0,0)$ to z_1, the subpath of Q_1 from z_1 to p_1, and the path Q. This contradicts Lemma 1.

Case 2: z_1 is on Q.

Here we consider subcases depending on the relative position of z_1 and z_0 on Q. Note that $z_0 \neq z_1$. For if $z_0 = z_1$, then ϕ is the identity translation, and hence, $p_0 = p_1$. This contradicts our assumption that tile r is repeated twice.

Case 2.1: z_1 precedes z_0 on Q (see Figure 2.4).

Then there is a cycle (from z_1 to p_1 along Q and back to z_1 along $\bar{Q}_1$) containing tile q twice; namely positions $\bar{z}_0$ and $\bar{z}_1$. This contradicts Corollary 3.

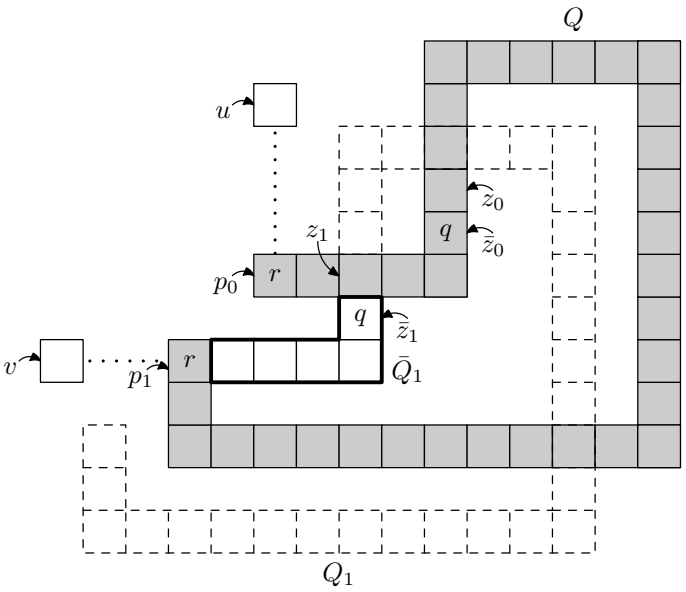


Figure 2.4: Intersection vertex z_1 precedes z_0 on Q. Then there is a path starting at $(0, 0)$ on which tile q appears twice.

Case 2.2 : z_1 succeeds z_0 on Q.

Let s_1 be the first vertex on $\bar{Q}_1$ starting from $\bar{z}_1$ that is also on P, and let $\bar{s}_1$ be the neighbour of s_1 on Q_1 closer to z_1. We distinguish two more cases depending on the position of s_1.

Case 2.2.1: s_1 is on the subpath of P from u to p_0 (see Figure 2.5).

Let P' be the path obtained from P by replacing the subpath of P from s_1 to z_1 with the subpath of Q_1 from s_1 to z_1. We will also think of P' as being directed, with u again being defined as the first vertex of P'. Since $A[F] \cup A[P]$ and $A[R] \cup A_1[\bar{Q}_1]$ are both assemblies of $\mathbf{T}$, by Observation 1, $B = A[F] \cup A[P] \cup A_1[Q_1]$ is also an assembly of $\mathbf{T}$. Therefore, B is a subconfiguration of the unique terminal assembly A. Thus, since P' is a subgraph of the backbone graph of B, P' is a also a subgraph of the backbone graph of A.

We will show that P' has fewer bad repetitions than P. Removing vertices from P cannot increase the number of bad repetitions. The only new tiles added to P' are the tiles of the subpath of $\bar{Q}_1$ from $\bar{s}_1$ to $\bar{z}_1$ (it is possible that this subpath is empty

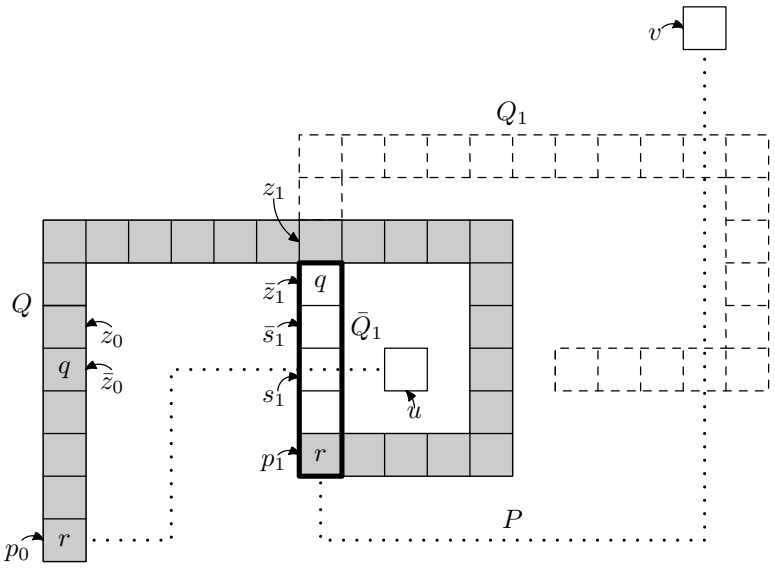


Figure 2.5: Intersection vertex z_1 succeeds z_0 on Q and s_1 lies on the subpath of P from u to p_0. Then replacing the sub-path of P from s_1 to z_1 (shaded) with the sub-path of Q_1 from s_1 to z_1 (unshaded) yields a path with fewer bad repetitions than P.

if $s_1 = \bar{z}_1$). These tiles appear on the part of P which is not included in P' (it is possible that these tile appear elsewhere on P as well). Moreover, each of these tiles is entered and left on P' via the same directions as the corresponding tiles on P, i.e., for every x on the subpath of P' from $\bar{s}_1$ to $\bar{z}_1$, $out^{P'}(x) = out^{P}(\phi^{-1}(x))$ and $in^{P'}(x) = in^{P}(\phi^{-1}(x))$. Hence, any bad repetition introduced by these new tiles only replaces a removed bad repetition involving corresponding tiles on the removed part of P (i.e., on the subpath of P from $\phi^{-1}(\bar{s}_1)$ to $\phi^{-1}(\bar{z}_1)$). Thus, P' does not have more bad repetitions than P. Next we show that tile r does not have a bad repetition on P'. Since z_0 precedes z_1 on Q, $p_0 \neq z_1$. Furthermore, s_1 precedes p_0 on P. Thus, p_0 is in between s_1 and z_1 on P, and hence not on P'. Therefore, tile r does not have a bad repetition on P'. Thus, P' has fewer bad repetitions than P.

Case 2.2.2: s_1 is on the subpath of P from p_1 to v (see Figure 2.6).

Note that it is possible that $s_1 = p_1$. Let P' be the path obtained from P by replacing the subpath of P from z_1 to s_1 with the subpath of Q_1 from z_1 to s_1. The only new tiles added to P' are the tiles of the subpath of $\bar{Q}_1$ from $\bar{s}_1$ to $\bar{z}_1$. These tiles already appear on P, in particular on its subpath Q. Note that these tiles are repeated exactly

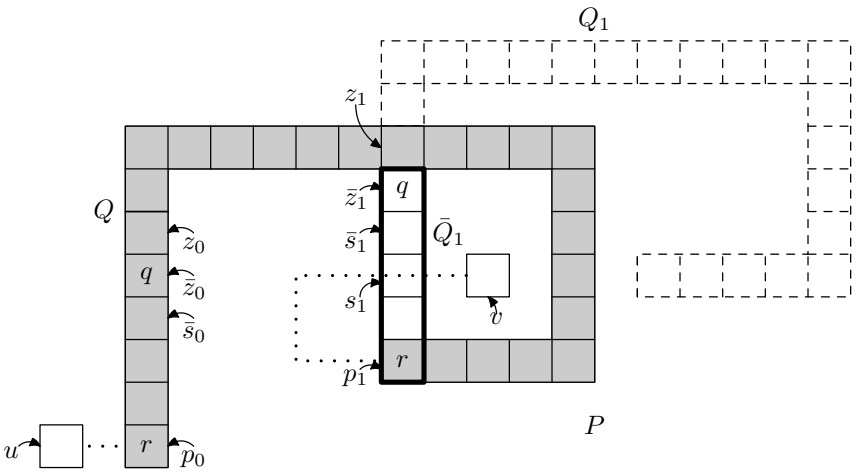


Figure 2.6: Intersection vertex z_1 succeeds z_0 on Q and s_1 lies on the subpath of P from p_1 to v. Then replacing the sub-path of P from z_1 to s_1 (shaded) with the sub-path of Q_1 from z_1 to s_1 (unshaded) yields a path with fewer bad repetitions than P.

twice on P', otherwise we would have a contradiction with Corollary 2. Each of the tiles on the subpath of P' from $\bar{z}_1$ to $\bar{s}_1$ is entered and left via opposite directions as the corresponding tiles of the subpath of P from $\bar{s}_0 = \phi^{-1}(\bar{s}_1)$ to $\bar{z}_0$, i.e., for every x on the subpath of P' from $\bar{z}_1$ to $\bar{s}_1$, $out^{P'}(x) = in^{P}(\phi^{-1}(x))$ and $in^{P'}(x) = out^{P}(\phi^{-1}(x))$. Hence, these repeating tiles create good repetitions, and no new bad repetitions were introduced on P'. Moreover, if $s_1 = p_1$, then by the argument above, tile r now has a good repetition on P'. If $s_1 \neq p_1$, then r is not repeated on P'. Therefore, P' has fewer bad repetitions than P, contradicting the minimality of P. This proves the lemma. □

Theorem 23. *Let* $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ *be a tile system that uniquely produces a finite assembly* A*. Suppose there are no binding domain mismatches in* A*. Then* T *contains at least* $\text{Mdiam}(A) + 1$ *non-empty tiles.*

Proof. Let u and v be two vertices in the backbone graph of A such that their Manhattan distance is $\text{Mdiam}(A)$. By Lemma 3 there is a path from u to v that does not contain any bad repetitions. Then by Lemma 2, T contains at least $\text{Mdiam}(A) + 1$ non-empty tiles. □

Applying the Theorem 23 to an $N \times N$ square, yields the following corollary.

Corollary 4. *Let* $\mathbf{T} = (\Sigma, T, C_t, s_\Sigma, 1)$ *be a tile system that uniquely produces an* $N \times N$ *square* A*. Suppose there are no binding domain mismatches in* A*. Then* T *contains at least* $2N - 1$ *non-empty tiles.*

This is a partial answer to an open question from [18]. The authors conjectured that $2N-1$ distinct tile types are required to uniquely assemble an $N \times N$ square at temperature 1. Corollary 4 shows if we assume that a tile system uniquely produces an $N \times N$ square *without any binding domain mismatches*, then $2N-1$ distinct tile types are indeed required.

The second bound (Theorem 23) gives a better lower bound compared to the first bound (Theorem 22) for an $N \times N$ square. However, for many uniquely produced assemblies Theorem 22 gives a better bound. For instance, for an S-shaped assembly, see Figure 2.7, Theorem 22 gives a lower bound of 24 while Theorem 23 gives a lower bound of 23. Obviously, this difference can be made arbitrarily large by increasing the width of the shape.

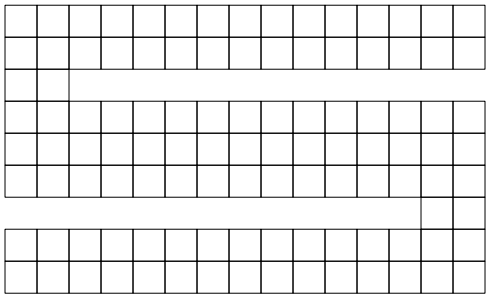


Figure 2.7: An example of a shape for which Theorem 22 gives a better lower bound (24 tile types) than Theorem 23 (23 tile types).

Chapter 3

The Step Assembly Model

Here we introduce a variant of the standard tile assembly model, called the "step assembly model". Roughly speaking, this model differs from the standard tile assembly model, in that we allow for several sets of tiles. We immerse a seed tile into one of the sets, "filter out" the assembled shape from this set and place this assembled shape into a different set of tiles where it now acts as a seed and assembly continues. We do not restrict the number of times a shape can be filtered out and placed in a new set of tiles, nor do we restrict the number of different sets of tiles. More formally, a *step tile system* is a 5-tuple $\mathbf{T}_{step} = (\Sigma, \{T_i\}_{i=1}^{k}, \{C_t\}, g, \{\tau_i\}_{i=1}^{k})$, where $\{T_i\}_{i=1}^{k}$ is a sequence of finite sets of tiles (each set including the tile *empty*), C_t is the initial seed configuration (consisting of the single tile t), g is a strength function, and $\{\tau_i\}_{i=1}^{k}$ is a sequence of temperatures. We define the (standard) tile system at step 1 as $\mathbf{T}_1 = (\Sigma, T_1, \{C_t\}, g, \tau_1)$ and the (standard) tile system at step i as $\mathbf{T}_i = (\Sigma, T_i, Term(\mathbf{T}_{i-1}), g, \tau_i)$ for $2 \leq i \leq k$. This allows us to view step tile systems as a sequence of standard tile systems.

We define the set of terminal assemblies of $\mathbf{T}_{step}$ as $Term(\mathbf{T}_{step}) = Term(\mathbf{T}_k)$. Given a configuration A, we say that the step tile system $\mathbf{T}_{step}$ *uniquely produces* A, if $\mathbf{T}_k$ uniquely produces A. We are interested in the number of tile types used in a tile system and define the *tile complexity* of $\mathbf{T}_{step}$ as $|\bigcup_{i=1}^{k} T_i| - 1$ (where we are subtracting 1 to exclude the *empty* tile).

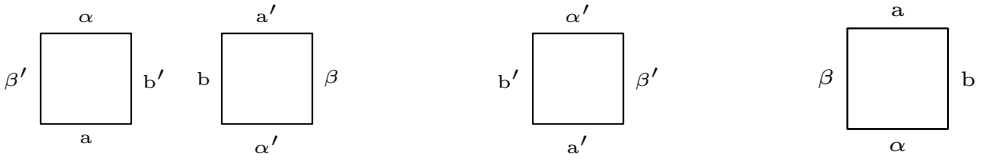
The step assembly model is similar to the staged assembly model proposed by Demaine et al. [7]. Recall that in the staged assembly model, there are any number of bins which

contain supertiles that self-assemble within the bins as in the multiple tile assembly model. During a stage, any collection of the following two types of operations can be performed: (i) add (arbitrarily many copies of) a new tile to an existing bin, (ii) pour one bin into another bin, mixing the contents of the former bin into the latter bin, and keeping the former bin intact. The key differences between our step assembly model and the staged assembly model are that in our model growth occurs by addition of single tiles to the seed configuration, while in [7], two assemblies are allowed to attach to form a larger assembly. Moreover, our model uses only one bin; we do not store assemblies for later mixing. At the end of this chapter, we will compare our results for tile complexity in the step assembly model to tile complexity results in the staged assembly model.

3.1 Tile Complexity of Full Squares at Temperature 1

Recall that in the standard tile assembly model (where only a single set of tiles is allowed), N^2 distinct tile types are required to uniquely assemble an $N \times N$ full square at temperature 1 [18]. Using the step assembly model, far fewer tile types suffice.

Example 1. *A full $N \times N$ square can be uniquely assembled using only 4 distinct non-empty tile types if N is odd, and 9 distinct non-empty tile types if N is even, under the step assembly model with temperature 1.*



(a) Tile set S_1. (b) Tile set S_2. (c) Tile set S_3.

Figure 3.1: The tile sets for assembling $N \times N$ squares, where N is odd.

Figure 3.1 shows the tile sets for assembling an $N \times N$ square, where N is odd. As we are working with temperature 1, all binding domains have strength 1. The tile from S_3 also

serves as the seed tile. The tile set sequence $\{T_i\}_{i=1}^{N-1}$ is defined as follows:

$$T_i = \begin{cases} S_1 & \text{if } i \text{ is odd,} \\ S_2 & \text{if } i \equiv 2 \pmod 4, \\ S_3 & \text{if } i \equiv 0 \pmod 4. \end{cases}$$

We further assume that each T_i contains the *empty* tile. Figure 3.2 shows the terminal assemblies for each step in the construction of a 5×5 square. Immersing the seed tile in the tile set S_1 causes the tiles of S_1 to attach to the northern, southern, eastern, and western boundary of the seed configuration, creating a "plus sign". Adding the tiles of S_2 to this terminal assembly causes the corners to fill in, creating a 3×3 square. In step $2k-1$, $k \geq 2$, the tiles of S_1 attach to every second tile on the northern, southern, eastern, and western boundary of the seed square. Immersing this assembly now into the tile set T_{2k} causes the tile from T_{2k} to fill in every other tile on the boundary of the new seed assembly, creating a $(2k+1) \times (2k+1)$ square. Thus, T_{N-1} uniquely produces a $N \times N$ square.

If we want to build an $N \times N$ square, where N is even, we first assemble an $(N-1) \times (N-1)$ square as above, and then use the tile sets E_1, E_2, and E_3 as illustrated in Figure 3.3. We first place the $(N-1) \times (N-1)$ square into the tile set E_1. The resulting shape is immersed into the tile set E_2, and finally the resulting terminal assembly is placed into E_3. (If the binding domains exposed on the north and west side of the $(N-1) \times (N-1)$ square are a, a' and b, b', then we make the following binding domain replacements to the tiles of E_1 and E_2: $\alpha \to a'$, $\alpha' \to a$, $\beta \to b'$, and $\beta' \to b$.) Figure 3.4 shows the assembly of a 4×4 square.

3.2 Tile Complexity of Arbitrary Shapes at Temperature 1

In this section we give an upper bound on the tile complexity of arbitrary shapes in the step assembly model at temperature 1. Recall that a shape S is a connected induced subgraph of the lattice. We say a configuration A *has shape* S, if $V(A) = V(S)$. We say a step tile system $\mathbf{T}_{step}$ *uniquely produces shape* S, if $\mathbf{T}_{step}$ uniquely produces assembly A and A has shape S. Before we can state the main result of this section, we need some more definitions.

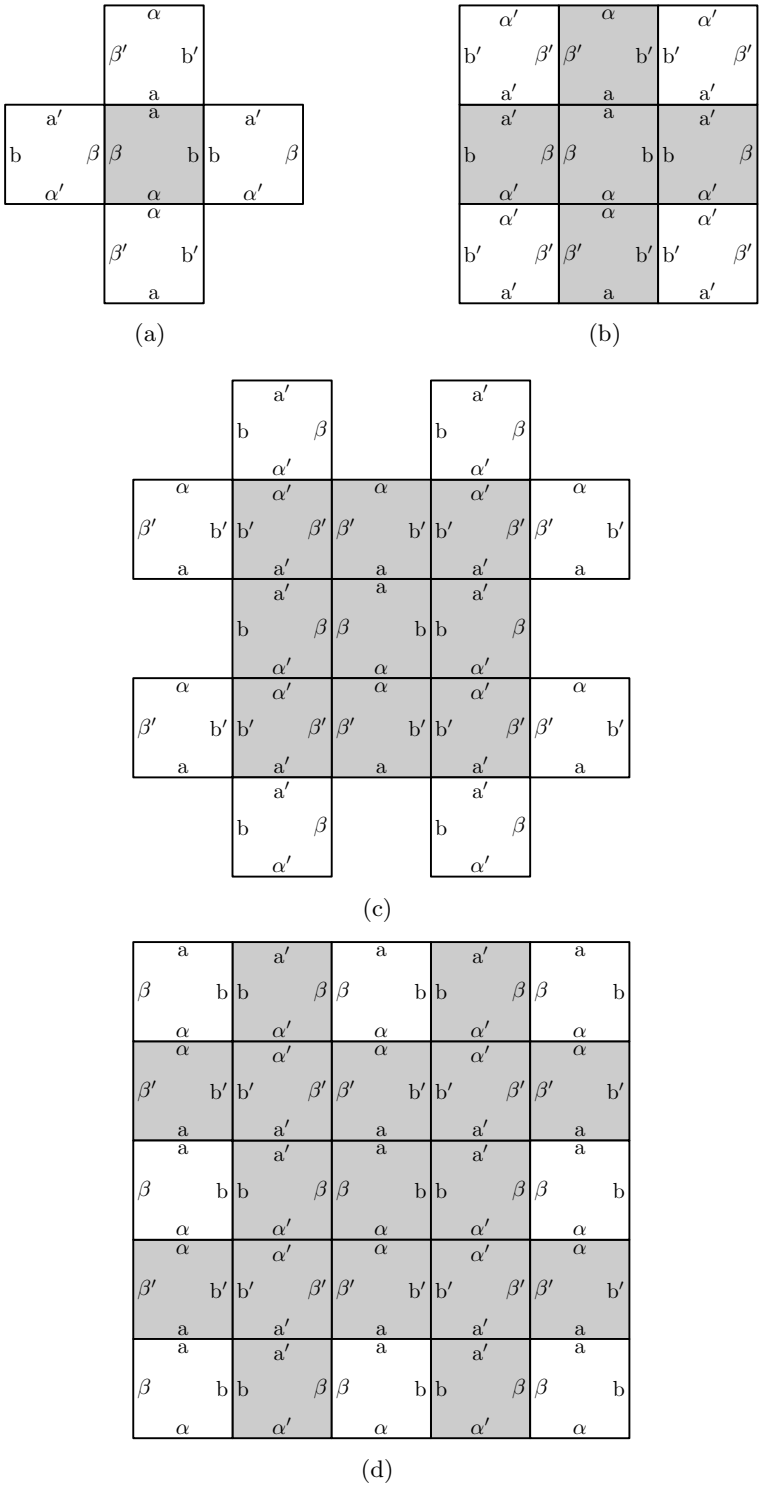
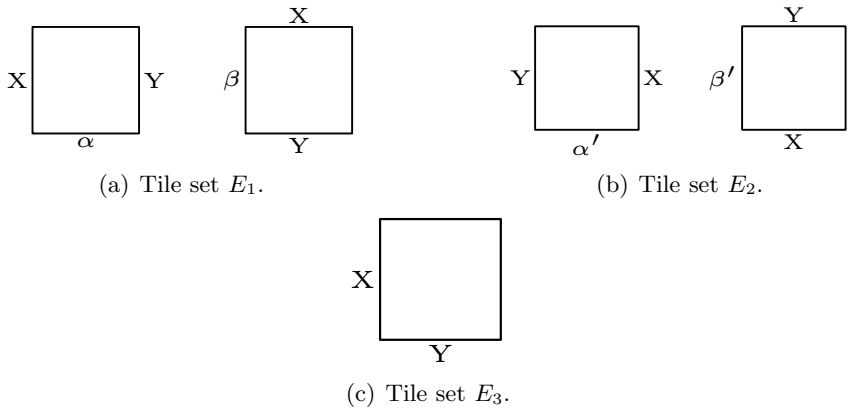


Figure 3.2: The terminal assemblies for each step in the construction of a 5×5 square. The seed configurations are indicated in grey.



(a) Tile set E_1.

(b) Tile set E_2.

(c) Tile set E_3.

Figure 3.3: Additional tile sets for assembling an $N \times N$ square, where N is even.

A *caterpillar* K is a graph consisting of a single path, called the *spine*, and additional vertices attached to the spine by single edges, called *hairs*. We refer to these additional vertices as *hairtips*. Given a tree F, a caterpillar K with spine P and set of hairtips L is a *natural* caterpillar if it is a subgraph of F and all vertices in L are leaves in F. Moreover, K is *maximal* if it is natural and the endpoints of P are leaves of F. If v is an endpoint of the spine P of the caterpillar K, we say that K is *anchored at* v. Moreover, K is a *maximal caterpillar anchored at* v, if K is a natural caterpillar anchored at v and the other endpoint of the spine (different from v) is a leaf of F.

Given a tree F, the sequence of sets of caterpillars $\{L_i\}_{i=1}^{\delta}$ forms an *m-level decomposition* D_F of F if it satisfies all of the following:

(mLD1) All caterpillars in $\bigcup_{i=1}^{\delta} L_i$ are edge disjoint, and cover all edges of F.

(mLD2) L_1 consists of a single maximal caterpillar of F.

(mLD3) For every $2 \leq i \leq \delta$, caterpillars in L_i are vertex disjoint.

(mLD4) For every $2 \leq i \leq \delta$, every caterpillar K in L_i is a maximal caterpillar anchored at a vertex v on the spine of some caterpillar in L_{i-1}. We call v the *anchor* of K.

Note that the anchor of K is either an internal vertex of the spine of a caterpillar K' or the anchor of K', where $K' \in L_{i-1}$. A vertex v has *level* i in D_F, if i is the smallest index such that v is a vertex of a caterpillar in L_i. The *depth of an m-level decomposition*

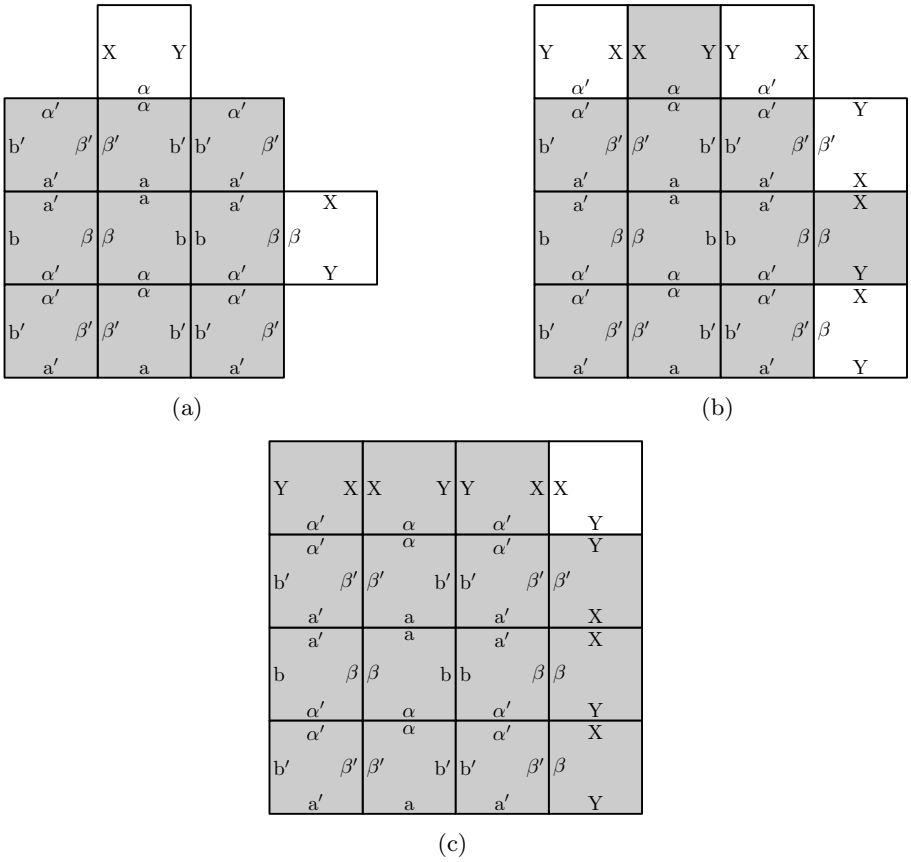


Figure 3.4: The terminal assemblies for the additional steps in the construction of a 4×4 square. The seed configurations are indicated in grey.

$D_F = \{L_i\}_{i=1}^{\delta}$ is δ. The *m-level depth of a tree* F, denoted by mLD-depth(F), is the smallest δ such that F has an m-level decomposition of depth δ. The *m-level depth of a shape* S, mLD-depth(S), is the minimum m-level depth of all spanning trees of S.

A vertex v is a *single anchor* if it is an anchor for exactly one caterpillar in D_F, and v is a *double anchor* if it is an anchor for exactly two caterpillars in D_F. Note that if F is a spanning tree of a shape S, then v cannot be an anchor for more than two caterpillars, since $deg_F(v) \leq 4$, and every anchor is an internal vertex of the spine of some caterpillar. Note that since for every $1 < i \leq \delta$ caterpillars of L_i are vertex disjoint, any vertex v of level j that is a double anchor is an anchor for one caterpillar of L_{j+1} and one caterpillar of L_{j+2}; moreover, $\deg_F(v) = 4$ and the remaining two edges incident to v belong to a caterpillar of L_j. Note that in this case v cannot be a leaf of the caterpillar of L_j.

Theorem 24. *Any shape S that has a spanning tree F of m-level depth δ can be uniquely produced by a step tile system at temperature 1 with tile complexity at most $68\delta + 8$ (using $2\delta + 2$ non-null binding domains). Moreover, if the maximum degree of F is 3, S can be uniquely produced by a step tile system with tile complexity at most $44\delta + 8$ (using $2\delta + 1$ non-null binding domains), and if the maximum degree of F is 2, only 14 tile types (and 2 non-null binding domains) suffice.*

Proof. Let $D_F = \{L_i\}_{i=1}^{\delta}$ be an m-level decomposition of F of depth δ. In the step tile system we are constructing, each tile set will contain only a single non-empty tile type. The binding domains will be taken from the set $\Sigma = \{null, a_1, a_2, \ldots, a_{\delta+2}, b_1, b_2, \ldots, b_\delta\}$. We define a partial function $\overline{x}$ on Σ as follows: $\overline{a_i} = b_i$ and $\overline{b_i} = a_i$, for $1 \leq i \leq \delta$. We also define a *rank* on Σ as follows: $\text{rank}(a_i) = i$, $\text{rank}(b_i) = i$, and $\text{rank}(null) = 0$. Our construction is guided by a depth-first search ordering of the vertices of F obtained as follows. Let s_0 be an endpoint of the spine of the caterpillar in L_1 (without loss of generality we assume that $s_0 = (0, 0)$). The sequence $\{s_j\}_{j=0}^{|S|-1}$ is obtained by a depth-first search on the vertices of F starting at the vertex s_0. At each branching point the depth-first search will choose the next vertex to visit as follows: choose a non-visited neighbour that is a leaf, or, if none of the non-visited neighbours are leaves, choose a non-visited neighbour of the highest level.

For every vertex s_j of F let $t^j = (t^j_N, t^j_E, t^j_S, t^j_W)$ be the tile that will be placed onto s_j. The tile sets used will be the sets $T_j = \{t^j, empty\}$. Next we specify the tile types t^j for $j = 0, \ldots, |S| - 1$ in this order. Note that s_1 is a neighbour of s_0. Let $d \in \mathcal{D}$ be the unique direction such that $s_1 = d(s_0)$. Set $t^0_d = a_1$ and $t^0_\alpha = null$, if $\alpha \neq d$.

For $j > 1$ and $\deg_F(s_j) = 1$, let s_k be the neighbour of s_j such that $k < j$. Let d be the direction such that $s_k = d(s_j)$. Set $t^j_d = t^k_{d^{-1}}$ (this ensures that the tile t^j can attach to the neighbouring tile t^k), and $t^j_\alpha = null$, if $\alpha \neq d$.

For $j > 1$ and $\deg_F(s_j) = 2$, let s_k be the unique neighbour of s_j such that $k < j$. Note that s_{j+1} is also a neighbour of s_j, and, because $\deg_F(s_j) = 2$, s_k, s_j, and s_{j+1} all belong to the same caterpillar in some L_i. Let d_1, d_2 be directions such that $s_k = d_1(s_j)$, and $s_{j+1} = d_2(s_j)$. Set $t^j_{d_1} = t^k_{d_1^{-1}}$ (this ensures that the tile t^j can attach to its neighbouring tile t^k), $t^j_{d_2} = \overline{t^j_{d_1}}$, and $t^j_\alpha = null$, if $\alpha \notin \{d_1, d_2\}$.

For $j > 1$ and $\deg_T(s_j) = 3$, we distinguish two cases:

(i) s_j is not an anchor:

Then s_j belongs to exactly one caterpillar K in L_i for some i. Note that s_{j+1} is a neighbour of s_j and it is a leaf. Let s_k and s_m be the other two neighbours of s_j such that $k < j$ and $m > j$ (see Figure 3.5(a)). Note that $s_m = s_{j+2}$ and that s_j and all its neighbours belong to the same caterpillar. Let d_1, d_2, d_3 be directions such that $d_1(s_j) = s_k$, $d_2(s_j) = s_{j+1}$, and $d_3(s_j) = s_m$. Set $t^j_{d_1} = t^k_{d_1^{-1}}$ (this ensures that tile t^j can attach to the neighbouring tile t^k), $t^j_{d_2} = a_{i+1}$ (hence using a new binding domain for hairs of K), $t^j_{d_3} = \overline{t^j_{d_1}}$, and $t^j_\alpha = null$, if $\alpha \notin \{d_1, d_2, d_3\}$.

(ii) s_j is an anchor:

Then s_j must be a single anchor, since only vertices of degree four can be double anchors. Let s_j be an anchor for a caterpillar K' in L_{i+1}. It follows that s_j also belongs to a caterpillar K in L_i (see Figure 3.5(b)). Let s_k, s_m, d_1, d_2, and d_3 be defined as in case (i). Note that s_{j+1} is a neighbour of s_j and its level is $i + 1$. Set $t^j_{d_1} = t^k_{d_1^{-1}}$ (this ensures that tile t^j can attach to the neighbouring tile t^k), $t^j_{d_2} = a_{i+1}$, (as the caterpillar K' will use a new pair of binding domains a_{i+1}, b_{i+1}), $t^j_{d_3} = \overline{t^j_{d_1}}$, and

$t^j_\alpha = null$, if $\alpha \notin \{d_1, d_2, d_3\}$.

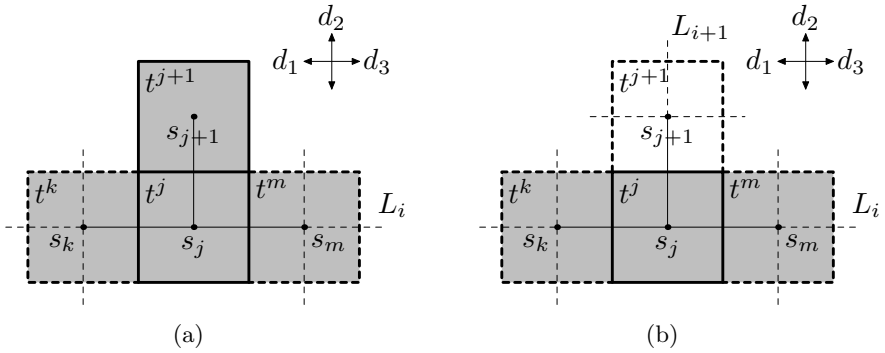


Figure 3.5: Tiles for a vertex of degree 3. The tiles t^n are indicated together with the vertices s_n of the spanning tree of S. Dashed sides of a tile indicate that another tile could attach to that side. Dashed edges of the tree indicate how the tree might continue. The shading represents the caterpillar of level i that the vertex s_j belongs to. The relative directions are indicated in the top right corner of each figure.

For $j > 1$ and $\deg_F(s_j) = 4$ we distinguish three cases:

(i) s_j is not an anchor:

Then two of the neighbours of s_j belong to the spine of some caterpillar K in L_i for some i, and the other two neighbours of s_j are leaves of K. Note that s_{j+1} is a neighbour of s_j and it is a leaf. Let s_k, s_m, and s_l be the neighbours of s_j different from s_{j+1} such that $k < j$, and $m > l > j$ (see Figure 3.6(a)), such that s_m is on the spine of K. Let d_1, d_2, d_3, d_4 be directions such that $d_1(s_j) = s_k$, $d_2(s_j) = s_l$, $d_3(s_j) = s_{j+1}$, and $d_4(s_j) = s_m$. Set $t^j_{d_1} = t^k_{d_1^{-1}}$ (this ensures that tile t^j can attach to the neighbouring tile t^k), $t^j_{d_2} = a_{i+1}$, $t^j_{d_3} = a_{i+2}$ (hence using new binding domains for the hairs incident to s_j), and $t^j_{d_4} = \overline{t^j_{d_1}}$.

(ii) s_j is a single anchor:

Let s_j be an anchor for a caterpillar K' in L_{i+1} for some i. It follows that s_j also belongs to a caterpillar K in L_i (see Figure 3.6(b)). Let s_k, s_m, s_l, and d_1, d_2, d_3, d_4 be defined as in case (i). Note that s_{j+1} is a neighbour of s_j and it is a leaf of K. Set $t^j_{d_1} = t^k_{d_1^{-1}}$ (this ensures that tile t^j can attach to the neighbouring tile t^k), $t^j_{d_2} = a_{i+1}$, (as the caterpillar K' will use a new pair of binding domains a_{i+1}, b_{i+1}), $t^j_{d_3} = a_{i+2}$ (hence using a new binding domain for the hair incident to s_j), and $t^j_{d_4} = \overline{t^j_{d_1}}$.

(iii) s_j is a double anchor:

Let s_j be an anchor for a caterpillar in L_{i+1} and a caterpillar in L_{i+2}. Let s_k, s_m, s_l, and d_1, d_2, d_3, d_4 be defined as in case (i). Since the neighbour s_{j+1} of s_j is visited before the neighbour s_l, s_{j+1} has a higher level than s_l. Therefore, s_{j+1} has level $i+2$ and s_l has level $i+1$. Set $t^j_{d_1} = t^k_{d_1^{-1}}$ (this ensures that tile t^j can attach to the neighbouring tile t^k), $t^j_{d_2} = a_{i+1}$ (as the caterpillar in L_{i+1} will use a new pair of binding domains a_{i+1}, b_{i+1}), $t^j_{d_3} = a_{i+2}$ (as the caterpillar in L_{i+2} will use a new pair of binding domains a_{i+2}, b_{i+2}), and $t^j_{d_4} = \overline{t^j_{d_1}}$.

Note that we based the assignment of binding domains for every tile t^j on the order in which the neighbours of s_j are visited by the search algorithm. That is, if s_l and s_m are two neighbours of s_j that both succeed s_j in the sequence obtained from the search algorithm, and s_l is visited before s_m, then the binding domain of tile t^j on the side adjacent to s_l has higher rank than the binding domain of t^j on the side adjacent to s_m.

Let the configuration C be defined by the map $C : \mathbb{Z} \times \mathbb{Z} \to \{t^0, t^1, \ldots, t^{|S|-1}\}$ such that

$$C(v) = \begin{cases} t^j & \text{if } v = s_j \text{ for some } s_j \in V(F), \\ empty & \text{otherwise.} \end{cases}$$

Clearly, the configuration C has shape S. We will show that the step tile system $\mathbf{T}_{step} = (\Sigma, \{t^j, empty\}_{j=1}^{|S|-1}, \{C_{t^0}\}, s_\Sigma, \{1\}_{j=1}^{|S|-1})$ uniquely produces C. To this end, let $C_j = C[V_j]$, where $V_j = \{s_0, \ldots, s_j\}$, be the sub-configuration of C induced by the first $j+1$ vertices ($j = 0, \ldots, |S|-1$). We say that the configuration C_j is *hierarchical*, if all of the following hold:

(H1) All non-*null* binding domains exposed on C_j have distinct ranks.

(H2) Let $x_1, x_2, \ldots, x_m$ be all exposed non-*null* binding domains on C_j ordered increasingly by their ranks. Let $t^{i_1}, t^{i_2}, \ldots, t^{i_m}$ be the corresponding tiles in C_j. Then $i_1 \leq i_2 \leq \cdots \leq i_m$.

We will show by induction that the tile system $\mathbf{T}_j$ of $\mathbf{T}_{step}$ uniquely produces C_j for each $j \geq 1$.

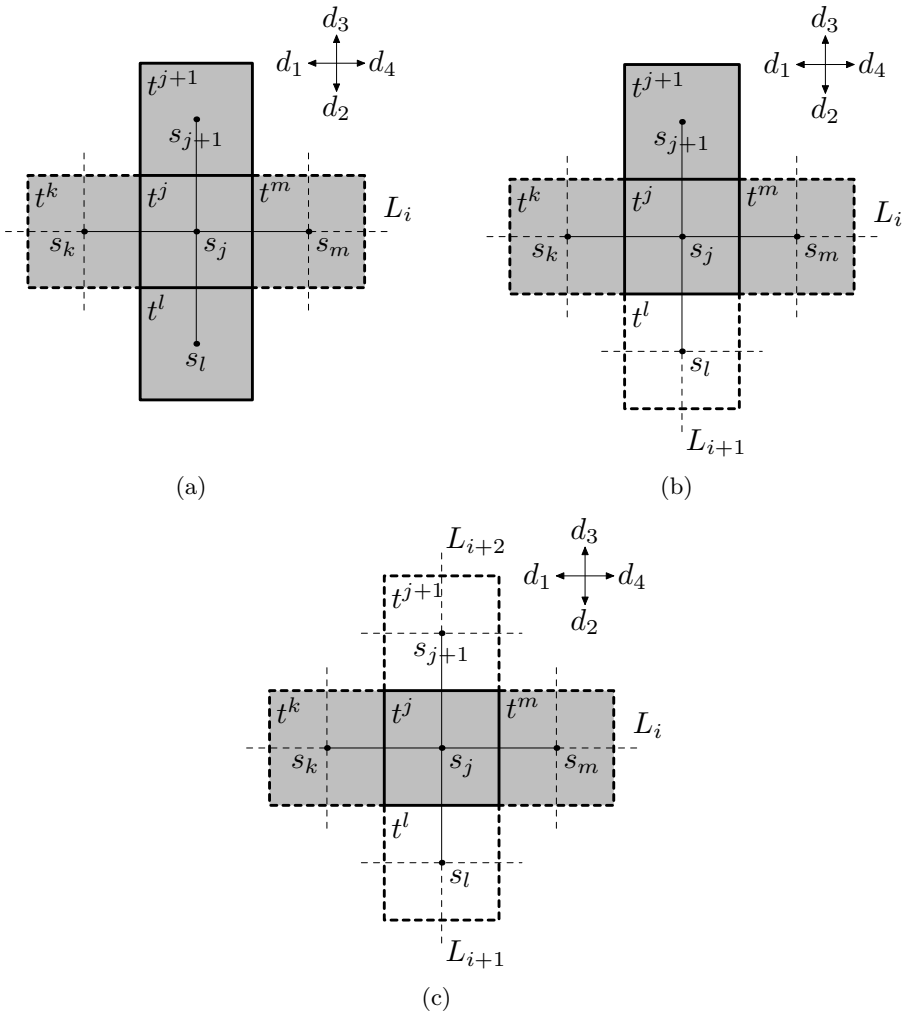


Figure 3.6: Tiles for a vertex of degree 4. The tiles t^n are indicated together with the vertices s_n of the spanning tree of S. Dashed sides of a tile indicate that another tile could attach to that side. Dashed edges of the tree indicate how the tree might continue. The shading represents the caterpillar of level i that the vertex s_j belongs to. The relative directions are indicated in the top right corner of each figure.

By definition of the step tile system $\mathbf{T}_{step}$, the seed configuration is $C_0 = C_{t^0}$. Since C_0 contains only one tile, condition (H2) is trivially satisfied. Since all non-*null* binding domains exposed on C_0 have distinct rank (as there is only one non-*null* binding domain, namely a_1, exposed), C_0 satisfies condition (H1). Hence, C_0 is hierarchical. Now we show that for every $j \geq 1$, if the set of seed configurations of $\mathbf{T}_j$ is $\{C_{j-1}\}$, and C_{j-1} is hierarchical, then $\mathbf{T}_j$ produces the unique terminal assembly C_j, and C_j is hierarchical. The claim will follow.

By the induction hypothesis, the set of seed configurations of $\mathbf{T}_j$ is $\{C_{j-1}\}$, i.e., $\mathbf{T}_j = (\Sigma, \{t^j, empty\}, \{C_{j-1}\}, s_\Sigma, 1)$. Let the vertex s_k be the neighbour of s_j in the tree F such that $k < j$. Since $C_{j-1}(s_j) = empty$, by our construction t^j can attach to C_{j-1} at position s_j in step j. Since t^j is the only non-*empty* tile in this step, it will attach at this position. Next we show that t^j cannot attach anywhere else. Let r be the rank of the binding domain via which tiles t^j and t^k are attached to each other in our construction. Due to our assignment of binding domains, r is a lowest rank of non-*null* binding domains on t^j (there could be two binding domains on t^j with lowest rank), and r is the highest rank of exposed binding domains on t^k. By the depth-first search, either $t^k = t^{j-1}$ or t^{j-1} is a leaf of F and the algorithm backtracked to t^k, in which case the tiles $t^{k+1}, \ldots, t^{j-1}$ do not have any exposed non-*null* binding domains. This implies that the binding domain of highest rank exposed on C_{j-1} is on t^k, since C_{j-1} is hierarchical. As r is the highest rank of exposed binding domains on t^k, it follows that r is the highest rank of exposed binding domains on C_{j-1}. Since all binding domains exposed on C_{j-1} have distinct rank and r is a lowest rank of non-null binding domains on t^j, tile t^j cannot attach anywhere else. Hence, $\mathbf{T}_j$ uniquely produces C_j.

It remains to show that C_j is hierarchical. Let $X = x_1, x_2, \ldots, x_m$ be the exposed non-*null* binding domains of C_{j-1} ordered by increasing rank. In C_j, tile t^j is attached to t^k via binding domain x_m which is of rank r. Since C_{j-1} satisfies (H1), and r is the lowest non-*null* rank of the binding domains on t^j, and the exposed non-*null* binding domains on t^j in C_j are distinct, it follows that C_j satisfies (H1). Let $Y = y_1, y_2, \ldots, y_k$ be the exposed non-*null* binding domains of C_j ordered by increasing rank. Since r is the highest rank of all non-*null* binding domains exposed on C_{j-1} and r is also the lowest rank of all non-*null* binding domains on tile t^j, in the sequence Y all non-*null* binding domains of t^j are at the end of the sequence. Since all the preceding elements are a subsequence of X, they satisfy

(H2). Moreover, since all new (if any) binding domains in Y come from tile t^j, C_j also satisfies (H2). Thus, C_j is hierarchical.

Hence the step tile system $\mathbf{T}_{step}$ uniquely produces the terminal assembly C, and therefore, $\mathbf{T}_{step}$ uniquely produces shape S.

It remains to show that the tile complexity of $\mathbf{T}_{step}$ is as required. There are four categories of tile types: tiles with exactly one, two, three, or four non-*null* binding domains. For the first type there are two choices for binding domain type (a_i or b_i, for some i), four choices for which side of the tile will be assigned the selected binding domain, $\delta + 2$ choices for the rank of a_i, and δ choices for the rank of b_i. Hence there are $4 \cdot (\delta+2) + 4 \cdot \delta = 2 \cdot 4 \cdot \delta + 8$ such tiles. For tiles with exactly two non-*null* binding domains, both binding domains must have the same rank, i.e. the binding domains must be a_i and b_i for some i. There are $4 \cdot 3$ ways of assigning these binding domains to the sides of a tile, and there are δ choices for i. Hence, there are $4 \cdot 3 \cdot \delta$ different tiles with exactly two non-*null* binding domains. Tiles with exactly three non-*null* binding domains have binding domains a_i, b_i, a_{i+1} for some i. There are $4 \cdot 3 \cdot 2$ ways of assigning these binding domains to the sides of a tile. There are δ choices for i. Thus, there are $4 \cdot 3 \cdot 2 \cdot \delta$ different tiles with exactly three non-*null* binding domains. Similarly, tiles with exactly four non-*null* binding domains have binding domains a_i, b_i, a_{i+1}, a_{i+2} for some i. There are $4 \cdot 3 \cdot 2$ ways of assigning these binding domains to the sides of a tile. Again, there are δ choices for i. Thus, there are $4 \cdot 3 \cdot 2 \cdot \delta$ different tiles with exactly four non-*null* binding domains. Therefore, the shape S can be assembled uniquely by a tile system with tile complexity of at most $2 \cdot 4 \cdot \delta + 8 + 4 \cdot 3 \cdot \delta + 4 \cdot 3 \cdot 2 \cdot \delta + 4 \cdot 3 \cdot 2 \cdot \delta = 68\delta + 8$.

If the maximum degree of F is three, then the tile complexity of $\mathbf{T}_{step}$ is at most $2 \cdot 4 \cdot \delta + 8 + 4 \cdot 3 \cdot \delta + 4 \cdot 3 \cdot 2 \cdot \delta = 44\delta + 8$, since there are no tiles with non-*null* binding domains on all four sides used in the construction.

If the maximum degree of F is 2, then F is a path and only two non-*null* binding domains are needed: a_1 and b_1. There are 12 tile types using both a_1 and b_1 (exactly once). In addition, two tile types having only one non-*null* binding domain are used for the endpoints of the path F. Thus, if the maximum degree of F is 2, the tile complexity of $\mathbf{T}_{step}$ is 14. □

Note that it is not necessary that m-level decompositions satisfy condition (mLD3), however we insist on this condition in order to optimize the tile complexity. For m-level decompositions that do not satisfy condition (mLD3), we can use a similar construction as in the proof of Theorem 24. In this case, a vertex of degree 4 that is a double anchor could be an anchor for two caterpillars K and K' that are both in L_i for some i. To ensure the tile system uniquely produces the desired shape, we cannot use binding domains a_i and b_i for the spine of both K and K'. Instead we introduce new binding domains c_i and d_i that we use for the spine of K and keep the binding domains a_i and b_i for the spine of K'. This implies that we need at most $136\delta + 8$ non-*null* binding domains.

3.3 Level Decompositions and Monotone Connected Node Search Number

Level decompositions of a tree are closely related to its monotone connected node search number. However, the definition of level decomposition is tailored to optimize the number of tile types used. In what follows we will show that if in an m-level decomposition we do not require caterpillars to be maximal, then this does not change the m-level depth of a tree. Moreover, if an addition we omit condition (mLD3) from the definition, i.e., we allow two caterpillars in the same level to share a common vertex (anchor), the "new" level depth corresponds to the monotone connected node search number of the graph.

Given a tree F, the sequence of sets of caterpillars $\{L_i\}_{i=1}^{\delta}$ forms a *level decomposition* D_F of F if it satisfies all of the following:

(LD1) All caterpillars in $\bigcup_{i=1}^{\delta} L_i$ are edge disjoint, and cover all edges of F.

(LD2) L_1 consists of a single natural caterpillar of F.

(LD3) For every $2 \leq i \leq \delta$, caterpillars in L_i are vertex disjoint.

(LD4) For every $2 \leq i \leq \delta$, every caterpillar K in L_i is a natural caterpillar anchored at a vertex v on the spine of some caterpillar in L_{i-1}. We call v the *anchor* of K.

Observe that this definition differs from the definition of m-level decomposition only that we do not require the caterpillars to be maximal in (LD2) and (LD4). As before, the

depth of a level decomposition $D_F = \{L_i\}_{i=1}^{\delta}$ is δ. The *level depth of a tree* F, denoted LD-depth(F), is the smallest δ such that F has a level decomposition of depth δ. We will show that these changes to the definition of level decomposition do not affect the level depth of a tree.

Lemma 4. *For any tree* F, mLD-depth(F) = LD-depth(F).

Proof. Since any m-level decomposition of F is also a level decomposition of F, it follows that LD-depth(F) $\leq$ mLD-depth(F). On the other hand, let $D_F = \{L_i\}_{i=1}^{\delta}$, where δ = LD-depth(F), be a level decomposition of F with the smallest number of caterpillars. We argue that D_F is also an m-level decomposition of F.

Towards a contradiction, suppose that D_F is not an m-level decomposition, i.e., either the caterpillar in L_1 is not maximal, or a caterpillar in a higher level set L_i, $i > 1$, is not maximal anchored at a vertex u on the spine of a caterpillar in L_{i-1}. Let this caterpillar be K. In both cases we can choose a vertex v of the spine of caterpillar K which is not its anchor u and is also not a leaf in F. By (LD1) and (LD4) there is another caterpillar K' in L_{i+1} which is anchored at v. Let us modify D_F as follows: First, append K' to K at v, and leave the resulting caterpillar in L_i. Second, remove K' from L_{i+1}. Let F_u be the tree F rooted at u, and $F_u[v]$ be the subtree of F_u rooted at v. By (LD1) every caterpillar is now completely either in $F_u[v]$ or out of $F_u[v]$. Third, move every caterpillar in $F_u[v]$ from its level set L_j to L_{j-1}.

This modification obviously does not violate (LD1) and (LD2). Before our modification, every caterpillar in $F_u[v]$ except K' was in a level greater than $i+1$. Since K and K' have been concatenated and since all caterpillars in $F_u[v]$ decreased their levels consistently, (LD3) and (LD4) also hold. Hence, the new decomposition is a level decomposition and it is easy to see that its depth is at most δ. This is a contradiction, since it has one less caterpillar than D_F. Hence D_F is an m-level decomposition and mLD-depth(F) $\leq$ LD-depth(F). Therefore, mLD-depth(F) = LD-depth(F). □

As we noted in section 3.2, condition (LD3) is used only to optimize tile complexity. In order to achieve equivalence between the level depth of a tree and its monotone connected

node search number we omit this condition and obtain a "general level decomposition".

Given a tree F, the sequence of sets of caterpillars $\{L_i\}_{i=1}^{\delta}$ forms a *general level decomposition* of F if it satisfies all of the conditions: (LD1), (LD2), and (LD4).

The *depth* of a general level decomposition $\{L_i\}_{i=1}^{\delta}$ is again δ, the number of sets in the decomposition. The *GLD-depth of a tree* F, denoted GLD-depth(F), is the smallest δ such that F has a general level decomposition of depth δ. The *GLD-depth of a shape* S is the minimum GLD-depth of all spanning trees of S.

Node searching is a variant of graph searching which was first introduced by Kirousis and Papadimitriou [11]. We are given a graph whose edges are all "contaminated", and a set of searchers. The goal is to obtain a state of the graph in which all edges are simultaneously "clear". In node searching, an edge becomes *clear* if both its endpoints are concurrently occupied by a searcher. An edge e becomes recontaminated, if there is a path from a contaminated edge to e and there are no searchers on this path. In node searching there are two basic operations, called "search steps": (i) place a searcher on a vertex, (ii) remove a searcher from a vertex.

A *search strategy* is a sequence of search steps that results in all edges of the graph being simultaneously clear. The smallest number of searchers for which a search strategy exists for a graph G is called the *node search number* $ns(G)$ of G. A search strategy is *monotone* if no recontamination ever occurs. A search strategy is *connected* if the set of clear edges always induces a connected subgraph. We are interested in search strategies that are both monotone and connected, and call the minimum number of searchers for which such a strategy exists for the graph G the *monotone connect node search number*, denoted by $mcns(G)$.

Edge searching is a version of graph searching where, in addition to the two search steps allowed in node searching, searchers can slide along an edge. Moreover, an edge uv becomes clear only when a searcher slides along the edge from endpoint u to endpoint v. The edge search number of a graph G is simply denoted by $s(G)$. In [5] a characterization of trees T whose monotone connected (edge) search number, $mcs(T)$, is at most k is given. This characterization is in terms of k-caterpillars, which are similar to our level-decompositions.

Furthermore, it was shown in [5] that there is a unique obstruction for the class of trees T such that $mcs(T) \leq k$. These results can easily be adapted to level-decompositions and monotone connected node search. To state the result, we first need a definition.

Let B'_k be the tree obtained from the complete binary tree of depth k (the distance from the root to the leaves) by joining each leaf l_i to a new vertex v_i. Let D'_k be the tree obtained by joining the three roots of three copies of B'_{k-1} to a unique new vertex r (see Figure 3.7). Furthermore, we say a graph H is a *minor* of a graph G, denoted $H \preceq G$, if H is isomorphic to a graph obtained from a subgraph of G by zero or more edge contractions.

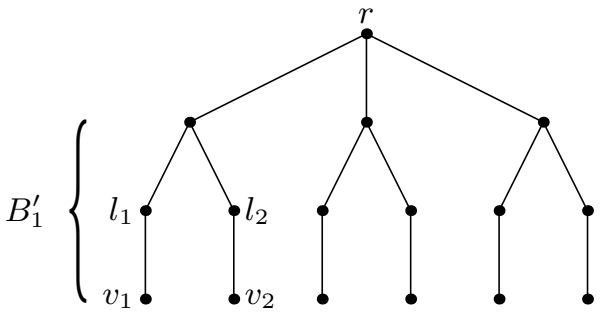


Figure 3.7: The tree D'_2.

Theorem 25. *For any tree F, the following three properties are equivalent:*

1. GLD-depth$(F) \geq k$;
2. $D'_{k-1} \preceq F$;
3. $mcns(F) \geq k+1$.

Proof. The proof of this theorem is analogous to the proof of Theorem 4.1 in [5]. We will use the following lemmas which we will prove later.

Lemma 5. *For any $k \geq 1$, $mcns(D'_k) \geq k+2$.*

Lemma 6. *For any tree F, such that* GLD-depth$(F) \leq k$, *$mcns(F) \leq k+1$.*

Lemma 7. *For any tree F, such that $D'_k \not\preceq F$,* GLD-depth$(F) \leq k$.

The proof of Theorem 25 now follows directly from these lemmas. If GLD-depth$(F) \geq k > k-1$, then by Lemma 7 we have $D'_{k-1} \preceq F$. Now suppose $D'_{k-1} \preceq F$. By Lemma 5 $mcns(D'_{k-1}) \geq k+1$, thus also $mcns(F) \geq k+1$. Finally, if $mcns(F) \geq k+1 > k$, then by Lemma 6 we have GLD-depth$(F) > k-1$. Hence, GLD-depth$(F) \geq k$. □

It remains to prove the lemmas.

Proof of Lemma 5. We will show that for any monotone connected node search strategy in D'_k, there is a step in which at least $k+2$ searchers are required to avoid recontamination. To this end, let T_1, T_2, and T_3 be the three subtrees attached to the root r of D'_k. Note that these subtrees are isomorphic to B'_{k-1}. Let i_1 be the first step during which an edge incident to the root r is cleared by a searcher. Without loss of generality, we may assume that T_1 and T_2 are still completely contaminated at step i_1. Let i_2 be the first step during which and edge e incident to a leaf l of T_1 or T_2 is cleared by a searcher. We may assume l belongs to T_1. At this point all other leaves of T_1 and all leaves of T_2 are still contaminated, while the path P from the leaf l to the root r is cleared (monotone connected search strategy). Searchers are located at the leaf l and its neighbour n (which is also a vertex of P), as otherwise edge e could not have been cleared during step i_2. For every other vertex x on the path P (including r), there is a path from x to a contaminated leaf. Since these paths are disjoint, each path needs to be guarded by a searcher. Thus, at least $k+2$ searchers are required during step i_2. Hence, $mcns(D'_k) \geq k+2$. □

Proof of Lemma 6. Let $\{L_i\}_{i=1}^{k}$ be a general level decomposition of F of depth k. Let the path P be the spine of the caterpillar in L_1. We show that there is a monotone connected node search strategy using $k+1$ searchers starting at one endpoint of P. We proceed by induction on k.

For the base case of $k = 1$, F is itself a caterpillar. Clearly, there exists a monotone connected node search strategy using 2 searchers starting at one endpoint of the spine of F. For the induction hypothesis, suppose that for every general level decomposition $\{L'_i\}_{i=1}^{\delta}$ of a tree F' of depth at most $k-1$, there exists a monotone connected node search strategy

starting at one endpoint of the spine of the caterpillar in L'_1, using at most $\delta+1 \leq k$ searchers.

For the induction step, let F be a tree with a general level decomposition $\{L_i\}_{i=1}^{k}$ of depth k. Let $P = \{v_0, v_1, \ldots, v_m\}$ be the spine of the caterpillar in L_1. For each vertex v_i denote by $w_{i,0}, w_{i,1}, \ldots, w_{i,d_i}$ the set of neighbours of v_i that are not in P. Denote by F_{v_i} the tree F rooted at v_i. Let $F_{v_i}[w_{i,j}]$ be the subtree of F_{v_i} rooted at $w_{i,j}$. Note that the tree $F_{v_i}[w_{i,j}]$ together with the edge $v_i w_{i,j}$, denoted $F_{v_i}[w_{i,j}] + v_i w_{i,j}$, is a tree that has a general level decomposition of depth at most $k-1$ where v_i is an endpoint of the caterpillar in the first set of this decomposition (for example, take the general level decomposition "induced" by $\{L_i\}_{i=1}^{k}$, i.e., $\{L_i \cap V(F_{v_i}[w_{i,j}] + v_i w_{i,j})\}_{i=2}^{k}$). We now use the following search strategy for F: place all $k+1$ searchers at v_0. Every time all searchers reach a new vertex v_i of P, leave one searcher at v_i and use k searchers to perform a monotone connected node search (starting from v_i) on the subtree $F_{v_i}[w_{i,j}] \cup v_i w_{i,j}$, for every $j = 0, \ldots, d_i$. Then move one searcher to v_{i+1} and leave the remaining k searchers at v_i to clear the edge $v_i v_{i+1}$. Now move the remaining searchers from v_i to v_{i+1}. This is a monotone connected node search strategy using $k+1$ searchers. Hence, $mcns(F) \leq k+1$. □

Proof of Lemma 7. We first need a definition. Given two trees F_1 and F_2 with roots x_1 and x_2, respectively, we say F_1 *is a x_2-rooted minor of* F_2, denoted by $F_1 \preceq_{x_2} F_2$, if F_1 is a minor of F_2 and vertex x_1 is either x_2 or the result of contracting a series of edges, some of which contain x_2 as an endpoint. As in the proof above, let $F_v[w]$ denote the subtree of tree F_v rooted at w, where F_v is the tree F rooted at v.

Let F be a tree, and let v be a vertex of F such that $B'_k \not\preceq_v F$. We will first show, by induction on k, that this implies that GLD-depth$(F) \leq k$ and that v is an endpoint of the spine of the caterpillar in the first set of a level decomposition of F of depth at most k. Then we show that for any tree F such that $D'_k \not\preceq F$, there exists a vertex v such that $B'_k \not\preceq_v F$.

For the base case of $k = 1$, it is easy to see that if $B'_1 \not\preceq_v F$, then F is a caterpillar and GLD-depth$(F) = 1$. Moreover, v is an endpoint of the spine of F. Indeed, assume that v has children $u_1, u_2, \ldots, u_s$ in F_v. If more than one of $u_1, u_2, \ldots, u_s$ has a child, then

$B'_1 \preceq_v F$. Otherwise, v can be chosen as an endpoint of a spine of F.

Now suppose the claim holds for some $k-1 \geq 1$, and there is a vertex v such that $B'_k \not\preceq_v F$. We consider two cases. First, if $B'_{k-1} \not\preceq_v F$, then by the induction hypothesis, GLD-depth$(F) \leq k-1 \leq k$. Second, if $B'_{k-1} \preceq_v F$, then let S be the set consisting of all vertices w such that $B'_{k-1} \preceq_w F_v[w]$. Since $B'_k \not\preceq_v F$, the vertices in S induce a path starting at v in F. Note that for each vertex $z \notin S$ such that z is adjacent to a vertex $w \in S$, $B'_{k-1} \not\preceq_z F_v[z]$. Hence, by induction, GLD-depth$(F_v[z]) \leq k-1$ and z is an endpoint of a spine of the caterpillar in the first set of a level decomposition of depth at most $k-1$. Now we can construct a general level decomposition of F as follows: The first set contains the caterpillar with S as its spine and all neighbours of the spine that are leafs in F are hairtips. The remaining sets of the decomposition are component-wise unions of optimal decompositions of $F_v[z]$ (where we extend the spine of every caterpillar in a first set to include one vertex of S), for every $z \notin S$ such that z has a neighbour in S. Thus, GLD-depth$(F) \leq k$, and v is an endpoint of the spine of the caterpillar in the first set of the decomposition.

It remains to show that there exists a vertex v such that $B'_k \not\preceq_v F$. Towards a contradiction, suppose that $D'_k \not\preceq F$ and for every vertex v of F, $B'_k \preceq_v F$. Then there exists a vertex z with two neighbours z_1 and z_2, such that $B'_{k-1} \preceq_{z_1} F_z[z_1]$ and $B'_{k-1} \preceq_{z_2} F_z[z_2]$. Since $B'_k \preceq_v F$ for every vertex v of F, this implies that either $B'_k \preceq_{z_i} F_z[z_i]$ or $B'_k \preceq_z F_{z_i}[z]$. In either case, we get that $D'_k \preceq F$, a contradiction. □

Rephrasing the equivalence of properties (1) and (3) we get:

Corollary 5. *For any tree F, $mcns(F) =$ GLD-depth$(F) + 1$.*

The connected node search number (where we do not insist that the search strategy is monotone) corresponds to another famous parameter in graph theory, namely the connected pathwidth. In [8] it is stated that the connected node search number of a graph is not always equal to the monotone connected node search number. However, in [8] it is also claimed that these two numbers are equal for trees, which implies that the GLD-depth of F is equal to the connected pathwidth of F.

3.4 Shapes with Constant Tile Complexity

In this section we exhibit a large class of shapes that have constant tile complexity under the step assembly model. The characterization of these shapes is based on "rectangle decompositions". Recall that a rectangle R is a shape for which there exist integers $N \geq 2$ and $M \geq 2$ and a vertex (x_0, y_0) such that vertex $(x, y) \in R$ if and only if $x_0 \leq x < x_0 + N$ and $y_0 \leq y < y_0 + M$. Given a shape S and an integer $k \geq 1$, we say S has a *rectangle decomposition* $\{R_i\}_{i=1}^m$ *with connectors of size* k if there exist m disjoint rectangles, $R_1, \ldots, R_m$, such that $V(S) = \bigcup_{i=1}^m V(R_i)$ and for each $i > 1$, there exists $j < i$ such that the rectangles R_i and R_j are joined in S by at least k edges. A selected set of k consecutive such edges is called the *connector* of R_i and R_j. Note that the endvertices of this connector in R_i (and R_j, respectively) induce a path of length $k-1$ which is called an *interface* of R_i (R_j). Observe that if R_i connects to l other rectangles it will have l interfaces.

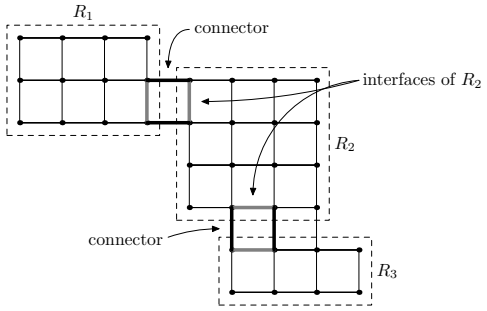


Figure 3.8: A rectangle decomposition of a shape with connectors of size 2. Connector edges are depicted in bold and interface edges are depicted in grey.

Theorem 26. *Under the step assembly model at temperature 1, any shape S which has a rectangle decomposition $\mathbf{R} = \{R_i\}_{i=1}^m$ with connectors of size 2, can be uniquely assembled using at most 24 non-empty tiles types.*

Proof. We will show that S has a spanning tree F of maximum degree at most 3 and level-depth at most 2. Given this spanning tree, we will obtain a tile system following the construction in the proof of Theorem 24. Due to our choice of F, this tile system will only use 24 non-*empty* tile types. Note that direct application of the result of Theorem 24 would give a bound of 96 non-*empty* tile types.

To obtain a suitable spanning tree F of S we will first show that S has a spanning subgraph G satisfying all of the following conditions:

(i) The maximum degree of G is 3.

(ii) G contains a cycle C that consists of all connector edges and all boundary edges of each rectangle in $\mathbf{R}$ which are not interface edges.

(iii) Every vertex not on the cycle C is on a "horizontal" path whose west endpoint belongs to C.

We will show this by induction on the number of rectangles in the rectangle decomposition $\mathbf{R}$ of S.

For the case when $m = 1$, S is itself a rectangle. Let C be the cycle consisting of all boundary edges of S. For every vertex w on the western boundary of S except the two corner vertices, let P_w be the path starting at w and going east with the other endpoint being the last vertex that is not on the boundary of S (see Figure 3.9). Then the cycle C together with all paths P_w forms a desired spanning subgraph G of S.

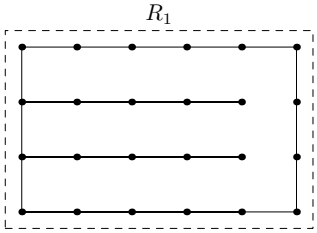


Figure 3.9: The spanning subgraph for the case of a single rectangle.

For the induction step, let S be any shape which has a rectangle decomposition $\mathbf{R} = \{R_i\}_{i=1}^{m}$ with connectors of size 2, for some integer $m \geq 2$. Let S' be the shape corresponding to the rectangle decomposition $\{R_i\}_{i=1}^{m-1}$. Since $\mathbf{R}$ has connectors of size 2 there exist two adjacent vertices u_m and v_m on the boundary of R_m such that for some $i < m$, u_m is adjacent to a vertex u_i of R_i and v_m is adjacent to a vertex v_i of R_i. By the induction hypothesis, since u_i and v_i are on the boundary of R_i, the edge u_iv_i is present in G'. Obtain a spanning subgraph G_m of R_m as described in the base case. Let C_m be the cycle of G_m. To obtain the cycle C for the spanning subgraph G of S, join C' and C_m by adding the connector edges

u_iu_m and v_iv_m, and deleting the interface edges u_iv_i and u_mv_m (see Figure 3.10). Now C together with the paths P_w from G' and G_m forms a spanning subgraph G of S. Since for every vertex $v \in G$, either $\deg_G(v) = \deg_{G'}(v)$ or $\deg_G(v) = \deg_{G_m}(v)$, and by the induction hypothesis, the maximum degree of G' and G_m is three, also the maximum degree of G is three. By the induction hypothesis, both G' and G_m satisfy condition (ii) for the rectangle decompositions $\{R_i\}_{i=1}^{m-1}$ and R_m, respectively. Since C contains the connector edges u_iu_m and v_iv_m, and all edges of C' and C_m except the edges u_iv_i and u_mv_m which are interface edges, also G satisfies condition (ii). By our construction, every vertex of G that is not on C is on a horizontal path P. This path is either in G' or G_m and thus, by the induction hypothesis, the west endpoint of P belongs to either C' or C_m. Since all vertices of C' and C_m belong to C, the west endpoint of P also belongs to C. Hence, G satisfies condition (iii).

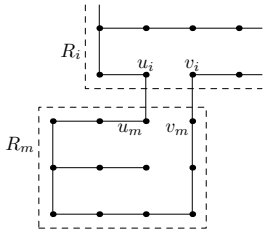


Figure 3.10: Extending the spanning subgraph to include R_m.

Given this spanning subgraph G of S, we obtain a spanning tree F of S by deleting from the cycle C an edge that is on the northern boundary of S. Certainly F will still have maximum degree three. It remains to show that F has level-depth at most 2. Let the caterpillar K consist of the path in F that we obtained by deleting an edge from the cycle C of G (this is the spine of K), together with any paths P_w that include only a single edge. Let $L_1 = \{K\}$. If $K = F$, then L_1 trivially forms a level decomposition of F. Otherwise let L_2 be the set consisting of all the paths P_w that are not included in the caterpillar K. Then all caterpillars in $L_1 \cup L_2$ are edge-disjoint and cover all edges of F. Hence, $\{L_1, L_2\}$ satisfies condition (LD1). Let u and v be the endpoints of the spine of K. Since u and v are on the northern boundary of S, they are leaves of F (any branching occurs on the western boundary of a rectangle). Thus K is a maximal caterpillar of F, and hence $\{L_1, L_2\}$ satisfies condition (LD2). By our construction, all paths P_w are vertex disjoint. Thus, $\{L_1, L_2\}$ satisfies condition (LD3). Every path P_w is a maximal caterpillar anchored at w, where w

is an internal vertex of the spine of caterpillar K in L_1. Hence, $\{L_1, L_2\}$ satisfies condition (LD4). Therefore, $\{L_1, L_2\}$ is a level decomposition of F, and hence F has depth at most 2.

Now construct a tile system from the spanning tree F of S as described in the proof of Theorem 24. However, due to our choice of F this tile system will use far less than the 96 tile types that Theorem 24 guarantees. Our tile system will use $4 \cdot 3$ tile types that use both a_1 and b_1 (exactly once) and no other non-*null* binding domains. There will be six tile types that use exactly three non-*null* binding domains (a_1, b_1, and a_2) (this is because any vertices of degree 3 are on the western boundary of a rectangle). There are two tile types that use both a_2 and b_2 (exactly once) and no other non-*null* binding domains, because all the paths in L_2 are horizontal. For the same reason, there are only two tile types with only one non-*null* binding domain which is taken from $\{a_2, b_2\}$. Furthermore, there will be two more tile types corresponding to the endpoints of the spine of the caterpillar of L_1 (both of these tiles have only one non-*null* binding domain - either a_1 or b_1). Therefore, our tile system uses only 24 non-empty tiles. □

If we are allowed to scale shapes by a factor of 2, then any shape can be assembled using a constant number of tile types.

Theorem 27. *Given an arbitrary shape S, let S' be the shape obtained by scaling S by a factor of 2. Then S' can be uniquely produced by a step assembly system at temperature 1 using at most 14 non-empty tile types.*

Proof. Each vertex of S corresponds to a 2×2 rectangle in S'. These rectangles from a rectangle decomposition of S' with connectors of size 2. Follow the proof above to obtain a spanning tree of S'. This spanning tree is in fact a path. Thus, Theorem 24 implies that 14 non-*empty* tile types and 2 non-*null* binding domains suffice to uniquely assemble S'. □

Given that the step assembly model is similar to the staged assembly model, it is interesting to compare tile complexity results under the different models. Both Theorem 27 of the step assembly model and Theorem 20 of the staged assembly model provide upper bounds on the tile complexity of uniquely assembling an arbitrary (non-full) shape at

temperature 1. Both constructions use 2 non-*null* binding domains. Theorem 27 uses 14 distinct non-*empty* tile types, while Theorem 20 uses 52 tile types. However, Theorem 27 only holds for shapes that are scaled by a factor of 2. Theorem 21 does not lend itself to comparison with our bounds in the step assembly model, as all of our constructions produce non-full assemblies, whereas Theorem 21 applies for full assemblies. For assembling $N \times N$ full squares, both models achieve constant tile complexity. The step assembly model has lower tile complexity than the staged assembly model, while the staged assembly model has a lower glue complexity than the step assembly model. Note that the focus in the staged assembly model was to minimize the glue complexity (rather than the tile complexity), while in the step assembly model we kept with the traditional complexity measure and focused on reducing tile complexity. Also note that while the two models are similar, they are not the same. In the step assembly model, assembly proceeds (as in the standard tile assembly model) by addition of single tiles, whereas in the staged assembly model two supertiles can attach to each other (as in the multiple tile model). Furthermore, while in the staged assembly model assembly takes place in several bins, the step assembly model only has a single bin.

Bibliography

[1] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. F. Knight, R. Nagpal, E. Rauch, G. J. Sussman, and R. Weiss. Amorphous computing. *Communications of the ACM*, 43:74–82, 2000.

[2] L. Adleman, Q. Cheng, A. Goel, M.-D. Huang, D. Kempe, P. M. de Espanes, and P.W.K. Rothemund. Combinatorial optimization problems in self-assembly. In *Proceedings of STOC*, pages 23–32, 2002.

[3] Leonard Adleman, Qi Cheng, Ashish Goel, and Ming-Deh Huang. Running time and program size for self-assembled squares. In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 740–748, 2001.

[4] 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 J. Comput.*, 34(6):1493–1515, 2005.

[5] L. Barriere, P. Fraigniaud, N. Santoro, and D. M. Thilikos. Connected and internal graph searching. In *29th Workshop on Graph Theoretic Concepts in Computer Science (WG), Springer-Verlag, LNCS 2880*, pages 34–45, 2003.

[6] Qi Cheng and Pablo Moisset de Espanes. Resolving two open problems in the self-assembly of squares. Technical Report 793, University of Southern Carolina, June 2003.

[7] E. D. Demaine, M. L. Demaine, S. P. Fekete, M. Ishaque, E. Rafalin, R. T. Schweller, and D. L. Souvaine. Staged self-assembly: nanomanufacture of arbitrary shapes with $O(1)$ glues. *Natural Computing*, 7(3):347–370, 2008.

[8] Pierre Fraigniaud and Nicolas Nisse. Monotony properties of connected visible graph searching. *Information and Computation*, 206(12):1383–1393, 2008.

[9] M. Gomez-Lopez, J. Preece, and J. Stoddart. The art and science of self-assembling molecular machines. *Nanotechnology*, 7:183–192, 1996.

[10] Ming-Yang Kao and Robert Schweller. Reducing tile complexity for self-assembly through temperature programming. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 571–580, 2006.

[11] L. Kirousis and C. Papadimitriou. Searching and pebbling. *Theor. Comput. Sci.*, 47(2):205–218, 1986.

[12] T.H. LaBean, H. Yan, J. Kopatsch, F. Liu, E. Winfree, J. H. Reif, and N.C. Seeman. Construction, analysis, ligation, and self-assembly of DNA triple crossover complexes. *Journal of the American Chemical Society*, 122:1848–1860, 2000.

[13] M. Lagoudakis and T. LaBean. 2D DNA self-assembly for satisfiability. In *Proceedings of the 5th DIMACS Workshop on DNA Based Computers*, pages 141–154, 1999.

[14] C. Mao, T. H. LaBean, J.H. Reif, and N.C. Seeman. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. *Nature*, 407:493–496, 2000.

[15] C. Mao, W. Sun, and N.C. Seeman. Designed two-dimensional DNA holliday junction arrays visualized by atomic force microscopy. *Journal of the American Chemical Society*, 121:5437–5443, 1999.

[16] J. Maňuch, L. Stacho, and C. Stoll. Two lower bounds for self-assemblies at temperature 1. In *Proc. of the 3rd international Conference on Bioinformatics and Biomedical Engineering (iCBBE, 2009)*. IEEE, 2009.

[17] J.H. Reif. Local parallel biomolecular computation. In *Proc. DNA-Based Computers*, pages 217–254, 1997.

[18] P. W. K. Rothemund and E. Winfree. The program-size complexity of self-assembled squares. In *Proceedings of STOC*, pages 459–468, 2000.

[19] Paul W. K. Rothemund. *Theory and Experiments in Algorithmic Self-Assembly*. PhD thesis, University of Southern California, 2001.

[20] P.W.K Rothemund, N. Papadakis, and E. Winfree. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biology*, 2:2041–2053, 2004.

[21] N.C. Seeman. DNA nanotechnology: novel DNA constructions. *Annual Review of Biophysics and Biomolecular Structure*, 27:225–248, 1998.

[22] D. Soloveichik and E. Winfree. Complexity of self-assembled shapes. *SIAM Journal on Computing*, 36:1544 – 1569, 2007.

[23] G. Whitesides, J. Mathias, and C. Seto. Molecular self-assembly and nanochemistry: a chemical strategy for the synthesis of nanostructures. *Science*, 254:1312–1319, 1991.

[24] E. Winfree, F. Liu, L. A. Wenzler, and N. C. Seeman. Design and self-assembly of two dimensional DNA crystals. *Nature*, 394:539–544, 1998.

[25] E. Winfree, X. Yang, and N. Seeman. Universal computation via self-assembly of DNA: Some theory and experiments. In *Proceedings of the Second Annual Meeting on DNA Based Computers*, pages 191–214, 1996.

Index