The Complexity of Multiple Handed Self-assembly

David Caballero, Timothy Gomez(✉), Robert Schweller, and Tim Wylie

University of Texas Rio Grande Valley, Edinburg, TX 78539-2999, USA
{david.caballero01,timothy.gomez01,robert.schweller,
timothy.wylie}@utrgv.edu

Abstract. In this paper we study complexities for the multiple-handed tile self-assembly model, a generalization of the two-handed tile assembly model in which assembly proceeds by repeatedly combining up to h assemblies together into larger assemblies. We first show that there exist shapes that are self-assembled with provably lower tile type complexities given more hands: we construct a class of shapes S_k that requires $\Omega(\frac{k}{h})$ tile types to self-assemble with h or fewer hands, and yet is self-assembled in $\mathcal{O}(1)$ tile types with k hands. We further examine the complexity of self-assembling the classic benchmark $n \times n$ square shape, and show how this is self-assembled in $\mathcal{O}(1)$ tile types with $\mathcal{O}(n)$ hands. We next explore the complexity of established *verification* problems. We show the problem of determining if a given assembly is *produced* by an h-handed system is polynomial time solvable, whereas the problem of *unique assembly verification* is coNP-complete if the hand parameter h is encoded in unary, and coNEXP-complete if h is encoded in binary.

1 Introduction

In this paper we investigate the complexity of fundamental problems related to multiple handed self-assembly. The model is a tile self-assembly model where system components are 4-sided Wang tiles, and self-assembly proceeds by tiles combining non-deterministically, based on matching glue types, to build larger assemblies. The most studied tile assembly models are the aTAM [26] where tiles attach one-by-one to a growing seed assembly, and the 2HAM [6] where assemblies are produced by taking any two assemblies (one in each hand) and combining them to create a new producible assembly. Our focus here is a generalization of the 2HAM, called the k-HAM, which allows groups of up to k assemblies (one in each of up to k hands) to combine to create new stable assemblies. See [20] for a general survey of tile self-assembly, and [27] for a survey of intrinsic simulation in tile self-assembly, and [11,28] for an overview of algorithmic self-assembly and recent experimental implementations in DNA.

This research was supported in part by National Science Foundation Grant CCF-1817602.

I. Kostitsyna and P. Orponen (Eds.): UCNC 2021, LNCS 12984, pp. 1–18, 2021.
https://doi.org/10.1007/978-3-030-87993-8_1

Why Multiple Hands? The most fundamental version of self-assembly may be the basic 2-handed model as the statistical likelihood of multiple pieces combining within a short enough time to stabilize is small in many experimental models. However, such productions happen in practice- often referred to as undesirable *spurious nucleation reactions* [19]. At certain scales, it is useful to consider the landscape of *stable* configurations, especially *local minimum energy* configurations, even when reaching such configurations would require moving more than two things into place simultaneously. Multiple handed self-assembly provides a simple framework for exploring self-assembly phenomena that may have impacts in these scenarios. Notably, multiple handed self-assembly may prove to be a useful tool for designing robustness in self-assembly systems, e.g., designing a system guaranteed to work even if the number of hands is increased up to some value k. Applications of such robustness theory would be directly aided by understanding fundamental complexities of self-assembly with multiple hands.

There are also classes of shapes that may not be built efficiently without multiple hands such as certain fractals that rely on multiple shapes coming together simultaneously [8], and there are shapes that cannot be built as efficiently at lower temperatures. Thus, given different experimental constraints, a reliance on these interactions with fewer tiles at a lower temperature may be preferable.

Our Results. We provide many results that explore multiple handed assembly from two angles: the complexity of fundamental problems, and complexity separation between the 2HAM and the h-HAM related to certain classes of shapes. Results are shown compared to previous work in Tables 1 and 2 respectively.

The first set of results within this model involve exploring the complexity of the computational problems of *producibility*, asking whether a given system of tiles produces a given assembly, and the *unique assembly verification* (UAV) problem asking whether a given tile system uniquely produces a given assembly (i.e. all producible assemblies can continue to grow into the single provided target assembly). We show that the producibility problem is solvable in polynomial time, and that the UAV problem is coNP-complete if the parameter k (number of hands) is encoded in unary, and coNEXP-complete if k is encoded in binary. In particular, these hardness results hold for the standard 2D scenario with a $\mathcal{O}(1)$-bounded temperature parameter. In comparison, while the UAV problem is known to be coNP-complete for both the 3D 2HAM [6] and the 2D 2HAM for a non-constant bounded temperature [22], the complexity of UAV in the 2HAM for 2D $\mathcal{O}(1)$-bounded temperature is still open.

Our next set of results show that there exist shapes that can be built more efficiently with more hands. We first provide a class of shapes, S_k, such that S_k requires at least $\Omega(\frac{k}{h})$ unique tile types to be assembled for any system with at most h hands, and yet is buildable in $\mathcal{O}(1)$ tile types with a k-handed system. Next we consider the classic benchmark of an $n \times n$ square, and show this shape can be built with $\mathcal{O}(1)$ tile types and $\mathcal{O}(n)$-hands, which is provably fewer tile types than needed with a $\mathcal{O}(1)$-handed system for almost all integers n [21].

Related Work. The h-handed self-assembly model was introduced in [8] as a generalization of the 2HAM, and was used to build a version of the Sierpinski

Table 1. Tile complexity for $n \times n$ squares.

Model	Tile complexity	Thm.
aTAM/2HAM	$\Theta(\frac{\log n}{\log \log n})$	[21]
2HAM (high temperature)	$\mathcal{O}(2^{\log^* n})$	[23]
kHAM	$\Theta(1)$	Theorem 2

Table 2. Related and new results for verification problems. *Two results show coNP-hardness of UAV in the 2HAM, one uses a step into the third dimension and the other uses the temperature as part of the input.

Problem	Hands	Results	Thm.
Producibility	aTAM	P	[1]
Producibility	2	P	[12]
Producibility	k	P	Theorem 3
UAV	aTAM	P	[1]
UAV	2	coNP-Complete*	[6,22]
UAV	k (Unary)	coNP-Complete	Corollary 2
UAV	k (Binary)	NEXPTIME-Complete	Theorem 4

Triangle fractal. Earlier work compared 2-hands (2HAM) over a single hand (aTAM) and showed a provable gap in tile complexity for building certain shapes [6], coNP-completeness for the UAV problem in the 3D 2HAM with a constant temperature [6] and coNP-completeness in 2D for non-constant temperature [22], versus a polynomial time solution to UAV in the aTAM [1]. For the case of the *producibility* problem, both the 2HAM and aTAM have polynomial time solutions [1,12]. Later work considered *Unique Shape Verification*, showing coNP^{NP}-completeness for the 2HAM [24] and coNP-completeness for the aTAM [2].

In [5], the authors show a separation in the number of tile types needed to construct some shape between the deterministic aTAM (only one terminal assembly) and the non-deterministic version of the model (allows for multiple terminal assemblies all with the same shape).

Building infinite classes of shapes with a fixed size-$\mathcal{O}(1)$ set of tile types has been explored in several self-assembly models. For example, [15,25] show how a fixed tile set can be programmed to build general shapes by adjusting the temperature of the system over a sequence of stages. Similarly, [9] builds general shapes with a fixed set of tiles by mixing combinations of the tile set into different bins over a sequence of stages. In [7], arbitrarily large squares are self-assembled with a fixed tile set size by encoding the desired square width into the system temperature. In [14], large shapes are self-assembled with a fixed tile set by considering bonding functions between assemblies that require greater strength to hold together larger assemblies. In [4,10,16,18], fixed tile sets are used to

build general squares and shapes with high probability by encoding the desired target shape into the relative concentrations of the tiles within the system.

2 Definitions

Tiles. A *tile* is a non-rotatable unit square with each edge labeled with a *glue* from a set Σ. Each pair of glues $g_1, g_2 \in \Sigma$ has a non-negative integer *strength* $\text{str}(g_1, g_2)$.

Configurations, Bond Graphs, and Stability. A *configuration* is a partial function $A : \mathbb{Z}^2 \to T$ for some set of tiles T, i.e. an arrangement of tiles on a square grid. For a given configuration A, define the *bond graph* G_A to be the weighted grid graph in which each element of $\text{dom}(A)$ is a vertex, and the weight of the edge between a pair of tiles is equal to the strength of the coincident glue pair. A configuration is said to be τ-*stable* for positive integer τ if every edge cut of G_A has strength at least τ, and is τ-*unstable* otherwise.

Assemblies. For a configuration A and vector $\vec{u} = \langle u_x, u_y \rangle$ with $u_x, u_y \in \mathbb{Z}^2$, $A + \vec{u}$ denotes the configuration $A \circ f$, where $f(x, y) = (x + u_x, y + u_y)$. For two configurations A and B, B is a *translation* of A, written $B \simeq A$, provided that $B = A + \vec{u}$ for some vector $\vec{u}$. For a configuration A, the *assembly* of A is the set $\tilde{A} = \{B : B \simeq A\}$. An assembly $\tilde{A}$ is a *subassembly* of an assembly $\tilde{B}$, denoted $\tilde{A} \sqsubseteq \tilde{B}$, provided that there exists an $A \in \tilde{A}$ and $B \in \tilde{B}$ such that $A \subseteq B$. An assembly is τ-*stable* provided the configurations it contains are τ-stable. Assemblies $\tilde{A}$ and $\tilde{B}$ are τ-*combinable* into an assembly $\tilde{C}$ provided there exist $A \in \tilde{A}$, $B \in \tilde{B}$, and $C \in \tilde{C}$ such that $A \cup B = C$, $A \cap B = \varnothing$, and $\tilde{C}$ is τ-stable. Let the shape of an assembly A be the shape taken from the set of points in $dom(A)$.

Two-Handed Assembly. A Two-handed assembly system $\Gamma = (T, \tau)$ is an ordered tuple where T is the *tile set* and τ is a positive integer parameter called the *temperature*. For a system Γ, the set of *producible* assemblies P'_Γ is defined recursively as: 1) $S \subseteq P'_\Gamma$. 2) If $A, B \in P'_\Gamma$ are τ-combinable into C, then $C \in P'_\Gamma$.

k-Handed Assembly. The k-handed assembly model is a generalization of two-handed assembly model. A k-handed assembly system $\Gamma' = (T, k, \tau)$ is an ordered tuple where T is the *tile set*, k is the number of hands that can be used to produce an assembly, and τ is a positive integer parameter called the *temperature*. For a system Γ', the set of *producible* assemblies $P'_{\Gamma'}$ is defined recursively as follows:

1. $S \subseteq P'_{\Gamma'}$.
2. For $2 \leq k' \leq k$, if $\{A_1, A_2, \ldots, A_{k'}\} \subset P'_{\Gamma'}$ are τ-combinable into C, then $C \in P'_{\Gamma'}$.

A producible assembly is *terminal* provided it is not τ-combinable with any other producible assembly, and the set of all terminal assemblies of a system

Γ is denoted P_Γ. Intuitively, P'_Γ represents the set of all possible assemblies that can self-assemble from the initial set T, whereas P_Γ represents only the set of assemblies that cannot grow any further. The assemblies in P_Γ are *uniquely produced* iff for each $x \in P'_\Gamma$ there exists a corresponding $y \in P_\Gamma$ such that $x \sqsubseteq y$. Thus unique production implies that every producible assembly can be repeatedly combined with others to form an assembly in P_Γ.

Unique Production of Shapes and Assemblies. A system Γ uniquely assembles an assembly A if the system uniquely produces set P_Γ that contains only the assembly A and no other assemblies. In other words all producible assemblies can be combined to eventually form A. We say a system uniquely assembles a shape $\mathcal{S}$ if the system uniquely produces set P_Γ and for all $B \in P_\Gamma$, B has the shape $\mathcal{S}$.

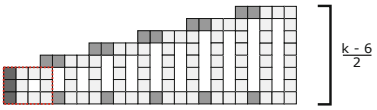


Fig. 1. Shape $\mathcal{S}_k$ is constructed by connecting width-3 loops of decreasing height starting at $\frac{k-6}{2}$. The base shape is highlighted by a red dotted box. Loops are shown in light gray. The darker column of 3 tiles on the left row is the cap column. The rest of the tiles are used to connect the loops. (Color figure online)

3 Shape Building

In this section we show a separation between systems with a differing number of hands. We start by defining a shape S_k and then proving the lower bound on the number of tiles needed to construct the shape in relation to the number of hands used, which is used to prove the separation.

3.1 Separation

We define the shape $\mathcal{S}_k$ for all even numbers $k \geq 14$ and for the smallest shape with $k = 11$. The shape for a given k, $\mathcal{S}_k$, is described in Fig. 1, and is built recursively in a $\tau = 3$ system. The smallest shape, $\mathcal{S}_{11}$, is highlighted in the figure and is a 3×3 loop with an additional 3 tiles on its left side which we will call the cap column. $\mathcal{S}_k$ is constructed by adding an additional height $\frac{k-6}{2}$ height loop on the left side of $\mathcal{S}_{k-1}$ and connecting it with 3 tiles (darker tiles in figure). Let $min_h(\mathcal{S}_k)$ be the minimum number of tile types needed to uniquely construct an assembly of shape $\mathcal{S}_k$ in an h-handed system.

Lemma 1. *For any h-handed system $\Gamma = (T, h, 3)$ that uniquely assembles the shape $\mathcal{S}_k$, $|T| \geq \Omega(\frac{k}{h})$*

Proof. Since Γ uniquely assembles the shape $\mathcal{S}_k$, each assembly in the set of terminals P'_Γ are of that shape. Consider the rightmost column of any assembly $A \in P'_\Gamma$, which we will call c, and let g be the number of strength $< \tau$ glues between tiles in this column. c can be divided into $g + 1$ segments that are connected using strength τ glues. Each segment is a producible assembly.

Since each of the segments have a strength τ (or greater) glue in between each other they cannot attach to the other segments unless a loop is formed. Let B be any producible assembly such that B is the subassembly of the shape S_k, but does not contain c. Since our system has h hands we are able to attach up to $h - 1$ segments to B in a single production step. Since the total length of the column is $\frac{k-6}{2}$, there must be a segment of length $\geq \frac{k-6}{2(h-1)}$. In order to build this segment, there must not be any repeated glues within that segment, otherwise the system could produce an infinitely growing assembly. Therefore, the number of tiles needed to construct this assembly is $\Omega(\frac{k}{h})$. □

3.2 Upper Bound for Building S_k

The tile set $T_\mathcal{S}$ is shown in Fig. 2a. Let the assembly A_k be an assembly of shape $\mathcal{S}_k$ shown in Fig. 2b.

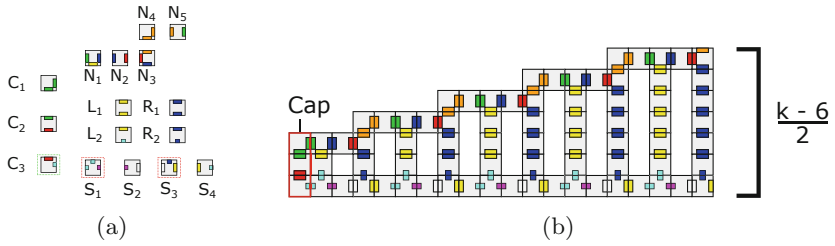


Fig. 2. (a) Constant sized tile set to construct an assembly with shape S_k with k hands. Larger rectangles represent glues of strength 2, while smaller rectangle represent strength 1. (b) Assembly of shape S_k made from the tile set.

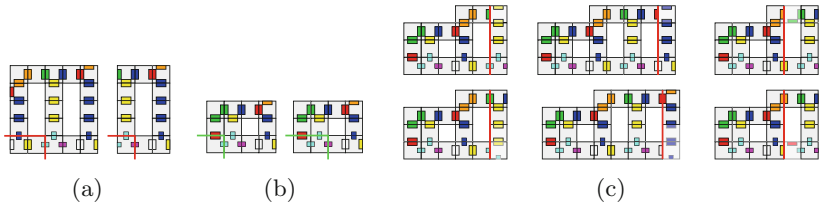


Fig. 3. (a) If one of these tiles are the bottom corner there will be a cut of strength 2 making the assembly not stable. (b) If the cap is on the assembly there does not exist a cut and the assembly is stable. (c) These are the possible conflicting tiles when attempting to construct a rogue assembly. The red line in each of these assemblies separates the column c from the rest of the assembly. (Color figure online)

Lemma 2. *Any τ-stable assembly in the h-handed system $\Gamma = (T_S, h, 3)$ is a subassembly of A_x for some $x > h$ and must contain the Cap column.*

Proof. Due to space constraints, we omit the proof and give the notes: 1) $\tau = 3$ and every glue is strength 1 or 2, so any stable assembly must contain a loop since any tile connected at only one point is not stable. 2) Only three tiles can be the bottom left corner of the loop (Fig. 2a). 3) There is a cut for all tiles unless the cap tile is present (Figs. 3a, 3b, and 3c). □

Lemma 3. *For all even $k \geq 12$ there exists a k-handed self-assembly system $\Gamma = (T, k, 3)$ uniquely assembling an assembly of shape $\mathcal{S}_k$ using $\mathcal{O}(1)$ tile types.*

Proof. Due to space constraints, this proof is not given, but we provide the tile set T in Fig. 2a as part of the system $\Gamma = (T, k, 3)$ that uniquely constructs the assembly seen in Fig. 2b using k hands. □

Theorem 1. *For all even $k \geq 12$ and $h < k$, there exists a shape $\mathcal{S}_k$ such that $min_h(\mathcal{S}_k) = \Omega(\frac{k}{h})$ and $min_k(\mathcal{S}_k) = \mathcal{O}(1)$. For the special case of $h = 2$, $min_2(\mathcal{S}_k) = \Omega(k)$.*

Proof. From Lemma 1, in the 2HAM the lower bound for constructing an assembly of shape S_k is $\Omega(k)$. From Lemma 3, the upper bound for uniquely constructing the shape is $\mathcal{O}(1)$ (Fig. 4). □

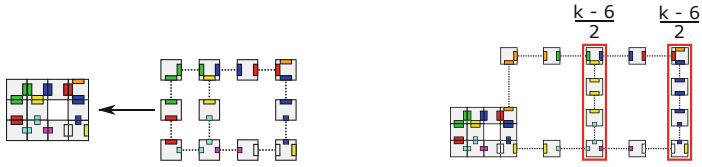


Fig. 4. (a) Using 11 hands, the base case of the assembly is built from single tiles. Using this as a single assembly, the next loop can be built. (b) For all future loops, they must be built by taking the previous sized assembly, the 5 tiles used to connect the two columns, and enough column tiles to connect them. This means that 6 hands are used to attach non-column tiles/assemblies and the remaining hands are used to build the two columns resulting in a max height of $\frac{k-6}{2}$.

3.3 Building Squares

In this section, we show that there exists a constant-sized tile set that can uniquely assemble the shape of an $n \times n$ square, where n is based on the given parameter specifying the number of hands of the system.

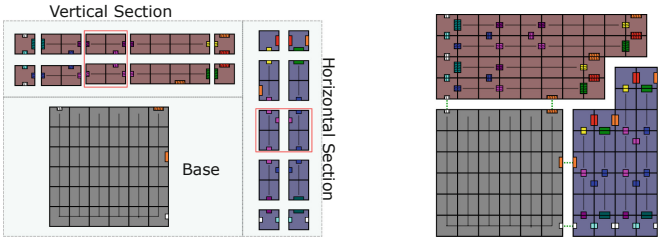
Theorem 2. *There exists a tile set T, consisting of 72 tile types, such that for all even integers $n \geq 10$, the h-handed tile assembly system $\Gamma = (T, h = n+1, \tau = 3)$ uniquely assembles an assembly A that has the shape of an $n \times n$ square.*

Proof. We prove by construction giving the tile set T (Fig. 5a). Solid lines represent unique strength-3 glues between the tiles. The tile set and final assembly consist of three sections. The base assembly is a 6×6 square that connects the other two sections. Both the horizontal and vertical sections build a staircase shaped structure, similar to Sect. 3.1, where each "step" of the staircase consists of a loop of tiles, and the largest buildable step is determined by the number of hands. This construction does not have space in the loops though, which creates rectangles increasing in size. We scale the size of the step by 2 and the vertical section 2 tiles taller in order for the three sections to fit together.

The tiles in the horizontal section build a staircase shaped assembly with increasing height. Attaching to the right, the tiles in the vertical section build a rotated staircase shaped assembly of increasing width as it builds upwards. This process continues until the addition of the next step requires more hands than allowed (Fig. 6). The two staircase assemblies and base assembly fit together to form a square shaped terminal assembly. An example is shown in Fig. 5b.

It is easily verifiable that in an h-handed system, an $(h-1) \times (h-1)$ square is producible. The two staircase assemblies are built up from the base assembly as shown in Fig. 6. The largest step of the horizontal assembly (blue) will be $h-3$, while the largest step of the vertical assembly (red) will be width $h-1$.

The argument that this tile system *uniquely* produces A is similar to that of Theorem 3. We focus on the horizontal section (blue) since the vertical section functions identically. In this case, the placement of tiles is even more restricted as the placement of the two repeating dominoes require each other to be stable due to the strength-1 glue between them.



(a) (b)

Fig. 5. (a) The tile set T that uniquely assembles an $(h-1) \times (h-1)$ square in an h-handed system. Solid lines between tiles represent a unique strength-2 glue between them. Small colored labels represent strength-1 glues, and large colored labels represent strength-2 glues. The glues in the vertical section are represented with the same colors as the horizontal section, but are hatched to signify they are distinct from their unhatched counterpart. The tiles boxed in red represent the section of each loop which can be repeated to make a loop of an arbitrary size. (b) This assembly is uniquely built in an 11-handed system. (Color figure online)

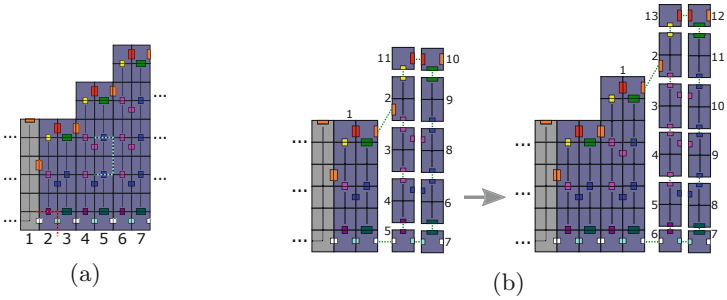


Fig. 6. (a) Cuts showing the assembly is unstable if column 1 or 3 are not the leftmost column in the horizontal section. The same cuts could larger in the assembly if there were no tiles to their left. (b) Construction of the largest horizontal staircase step in a 13-handed system. Note the largest step in the vertical section would be 12 tiles wide. (Color figure online)

A stable assembly must include the smallest step of the staircase. Figure 6a shows an assembly built from the tiles in the horizontal section. If column 2 or 5 were the leftmost column of this assembly, the red and cyan cuts show the resulting assembly is not stable, respectively. It is inherent that a stable assembly built from these tiles is either attached to the base assembly, or to column 3.

In a similar argument to the previous construction, only a few conflicting tiles exist that any rogue assembly may contain with the repeating dominoes, since without them it is clearly a subassembly of A. By starting from a pair of adjacent repeating dominoes, we work upwards and downwards, noting the possible tiles that *could* be placed, and see that they must exist in the "loop" composed of all the tiles of the horizontal section in order to be in a stable assembly. □

4 *h*-Hand Producibility

Here we show that verifying producibility of an assembly is solvable in the h-handed model in polynomial time. The proof is a modification of the proof of 2-handed producibility in [12] generalized to h-handed assembly. A partition of a configuration C is a set of unique configurations $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ such that $\bigcup_{i=1}^{h} C_i = A$ and for all $i \neq j$, $C_i \bigcap C_j = \emptyset$. With regards to a partition of an assembly A, we mean the partition of an arbitrary configuration $C \in A$.

Definition 1 (h-handed Assembly Tree). *An h-Handed Assembly Tree for a configuration C is a tree Υ where the root represents C, every other node represents a configuration $c \subseteq C$, every parent node has at most h children, and every parent node p has the characteristic that it's children are τ-combinable in an h-handed assembly step into p.*

Lemma 4. *For any h-handed system $\Gamma = (T, h, \tau)$ and partition C of assembly $A \in P'_\Gamma$, if $\forall a \in C, a \in P'_\Gamma$ then there exists a subset s of C such that $2 \leq |s| \leq h$ and the elements of s are τ-combinable into an assembly $B \sqsubseteq A$.*

Proof. Since $A \in P'_\Gamma$, there must exist an h-handed assembly tree Υ. We utilize a method from [12] to mark nodes in Υ to find a valid candidate assembly B. The previous proof used a more generalized version of an assembly tree.

Label each leaf $\{x\}$ in Υ with the unique element $C_i \in \mathcal{C}$ where $x \in C_i$. Then iteratively, if all siblings have the same label, label the parent C_i as well. This preserves the partition labels for each parent as long as it is a proper subset of the partition.

Doing a breadth first search from the root looking at only unlabeled vertices, we reach one of the 3 following cases,

1. The children have $2 \leq b \leq h$ distinct labels and there are b children.
2. The children have $2 \leq b \leq h-1$ distinct labels and there are $3 \leq c \leq h$ children and $c > b$.
3. There are labeled and unlabeled children or all unlabeled children. We ignore these nodes since there must exist nodes from either Case 1 or 2 if we follow the unlabeled children since all leaves are labeled.

Case 1. This case shows that there must exist b partitions that neighbor each other in the tree and can be brought together with b hands. There is a subtle subcase that for the parent assembly node p, some number of the children could be combined with fewer hands. However, a modification to the build path in this way does not change p since it could be built from the single b-handed operation or through multiple joins of less than b hands since each operation is joining subsets of different partitions and p can still be formed as a stable assembly.

Case 2. Even though $3 \leq c \leq k$ hands are needed, some of the nodes have the same label. Thus, the number of distinct partition subsets is $2 \leq b \leq h-1$. Similar to Case 1, some of the children could be combined without assembling all b children at once. Any stable combination of children represents another valid h-handed assembly sub-tree for the parent node.

Let s' be the set of configurations represented by the children of the found node. For each element in s' replace it with the element of C it was labeled by (only once for each label) to form the set s. This replacement preserves the ability for all the assemblies in s to be combinable. Since we know each element

Result: Given an h-handed assembly system $\Gamma = (T, h, \tau)$, and an assembly A, is A producible by Γ?
`/* Subassemblies of` A `as positions. Initially individual tiles. */`
$\mathcal{C} \leftarrow \{\{v\} | v \in dom(A)\}$;
while $|C| > 1$ **do**
  if $\exists\ 2 \leq b \leq h$ *subassemblies in* $\mathcal{C}$ *(denoted* $C_i \in \mathcal{C}$ *with* $1 \leq i \leq b$*) s. t.* $\cup_{1 \leq i \leq b} C_i$ *is stable* **then**
    $\mathcal{C} \leftarrow \mathcal{C} \setminus \{C_1, \ldots, C_b\} \cup \{\cup_{1 \leq i \leq b} C_i\}$
  else reject
accept;

Algorithm 1: The naïve method of verifying whether an assembly is producible in an h-handed system.

of s are producible, the assembly B is producible where the elements of s are τ-combinable into B. □

Theorem 3. *The producibility problem for a system $\Gamma = (T, h, \tau)$ and assembly A is solvable in $\mathcal{O}(|A|^2 h \log |A|)$ time.*

Proof. Algorithm 1 gives the naïve method for building the shape by combining tiles from the shape whenever possible. We know from Lemma 4 that if the target assembly A is producible, there must exist up to h subassemblies that may be combined at each step.

The runtime is affected by the time required to find cycles in planar graphs. In order for assemblies to be combined they must be adjacent. Any assembly step that requires more than 2 hands must form a loop. Thus, the bottleneck is checking for a cycle of size h in a planar graph, which varies based on h. We must also check for cycles up to size h, so we might require h calls to this subroutine. Let T be the time to find a cycle, then the runtime of Algorithm 1 is $\mathcal{O}(Th|A|)$.

Arbitrary fixed length cycles in planar graphs with n nodes can be found in $\mathcal{O}(n \log n)$ time (with expected $\mathcal{O}(n)$ time), and for any $h \leq 6$, the complexity is $\mathcal{O}(n)$ [3,17]. Thus, for an unknown h, the runtime of Algorithm 1 is $\mathcal{O}(|A|^2 h \log |A|)$ as the size of the graph is the size of the assembly.

We note that there is a $\mathcal{O}(n)$ algorithm to find any fixed subgraph H in a planar graph, but it requires an extremely large constant that is generally considered impractical even for small n [13]. Also in the special case of 2-handed assembly the runtime of the algorithm shown in [12] runs in time $\mathcal{O}(n \log^2 n)$. □

5 *h*-Hand Unique Assembly Verification

In this section we investigate the complexity of the problem of verifying an assembly is uniquely assembled by a given h-HAM system. We consider two different methods of encoding the number of hands in the system. We show that the problem is coNEXP-complete and coNP-complete when the number of hands is encoded in binary and unary, respectively. The problems are listed below. We first show membership, then prove hardness with a reduction from k-A_{NTM}.

*Problem 1 (*h-Ham-UAV*).* **Input:** An h-HAM system $\Gamma = (T, h, \tau)$ where the integer h is encoded in binary, and an assembly A. **Output:** Does Γ unique produce the assembly A?

*Problem 2 (*u-h-Ham-UAV*).* **Input:** An h-HAM system $\Gamma = (T, h, \tau)$, where the integer h is encoded in unary, and an assembly A. **Output:** Does Γ unique produce the assembly A?

5.1 Membership

The UAV problem is in the class coNEXP if the number of hands is specified in binary, and in coNP if is specified in unary. For an instance of UAV (Γ =

$(T, h, \tau), A)$, the instance is true if and only if the following 3 conditions are true: 1) The target assembly A is producible, 2) there does not exist a terminal assembly $C \sqsubset A$, and 3) there does not exist a producible assembly $B \not\sqsubseteq A$.

Lemma 5. H-HAM-UAV $\in$ *coNEXP.*

Proof. We provide a coNEXP algorithm for H-HAM-UAV that checks the above three conditions. By Theorem 3, condition 1 can be decided in polynomial time. Utilizing Lemma 4, we show that if condition 1 is true then condition 2 is true. For some assembly $C \sqsubset A$, consider any partition of the assembly A where C is an element, and by repeatedly applying Lemma 4, continue joining elements of this partition until A is built. To do this, C must, at some point, attach to another assembly. Therefore, C is not terminal.

The remaining task to decide the instance of UAV is to verify that there does not exist a producible assembly $B \not\sqsubseteq A$. A coNEXP machine can do this by nondeterministically attempting to build an assembly up to size $h|A|$. If any branch builds some assembly $B \not\sqsubseteq A$, then the branch (and machine) will reject. It suffices to check only up to this size, as an assembly of size $> h|A|$ must have been built from at least one assembly of size $> |A|$. That assembly itself is not a subassembly of A, and therefore if it exists, then a different branch of the computation will build it and reject. Since h is encoded in binary it takes exponential time to build an assembly of size $h|A|$. □

Corollary 1. U-H-HAM-UAV $\in$ *coNP.*

Proof. The proof of this is the same algorithm provided in Lemma 5. Since the integer h is encoded in unary, nondeterministically building an assembly of up to size $h|A|$ takes polynomial time. □

5.2 Hardness: Reduction from k-A$_{NTM}$

To show coNEXP-hardness we reduce from the canonical complete problem for NEXP, K-A$_{NTM}$, which is the problem of deciding if there exists a computation path of length $\leq k$ where a given nondeterministic Turing machine M accepts when run on the empty tape. We first overview the construction, and then prove correctness. The formal problem definitions follow.

*Problem 3 (*K-A$_{NTM}$*).* **Input:** A nondeterministic Turing machine M, and an integer k encoded in binary. **Output:** Does there exist a computation path of M accepting within k steps when run on an empty tape?

*Problem 4 (*U-K-A$_{NTM}$*).* **Input:** A nondeterministic Turing machine M, and an integer k encoded in unary. **Output:** Does there exist a computation path of M that accepts within k steps when run on an empty tape?

Given an instance of K-A$_{NTM}$, we create an instance of H-HAM-UAV such that the system is temperature-4, and the number of hands h is set to $(2^{\lceil \log_2(k) \rceil} + 3) \cdot (k + \lceil \log_2(k) \rceil + 4)$. The system always builds a specific target assembly. If and

only if the answer to K-A_{NTM} is yes, the system also produces a *computation assembly*, an assembly that represents an accepting computation path of M that is less than k steps. The computation assembly is *not* the target assembly, making the answer to the UAV instance 'no.' We will walk through an example, reducing from an instance $(M, k = 4)$ and creating a temperature-4 system where the number of hands is set to $(4 + 3) \cdot (4 + 2 + 4) = 70$.

We first explain the case when the instance of K-A_{NTM} is true, and therefore the created instance of H-HAM-UAV is false. In this case, a computation assembly is built. A computation assembly is composed of a binary counter section and a Turing machine simulation section (Fig. 7c). Since there exists an accepting computation path of less than or equal to 4 steps, then in one production step, utilizing the large number of hands in the system, ≤ 70 tiles will come together forming a tableau that represents a simulation of the computation path, as well as a binary counter enforcing the simulation to maintain a certain tape width.

Binary Counter. We utilize known techniques, such as in [21], for implementing a self-assembling binary counter where the construction has a size-$\mathcal{O}(\log_2 k)$ tile set and bounds the counter such that it stops once it reaches $2^{\lceil \log_2(k) \rceil}$. For our example instance $(M, k = 4)$, the assembled binary counter is shown in Fig. 7a. This assembly is one of two parts of the computation assembly, and is not stable by itself at temperature 4. The binary counter counts from left to right, starting at 0 and ending at $2^{\lceil \log_2(k) \rceil}$. The bottom row of light gray tiles represents the least significant bit, while the top row represents the most significant bit. Each row uses a distinct set of tiles, preventing unbounded growth. The majority of the tiles (light gray) have a strength-1 glue on each side. Thus, these tiles are adjacent to a tile on every side in order to be stable in the assembly. The remaining border tiles (dark gray) are the only tiles that can be on the border of a stable assembly due to their strength-2 glues. Every tile in the top row is not connected to the rest with the required strength of 4. As depicted, the assembly can only be stable in combination with an additional assembly above it.

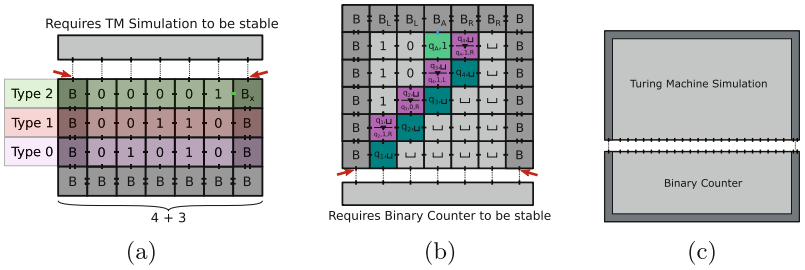


Fig. 7. (a) Example binary counter that counts up to 4. Each type consists of its own $\mathcal{O}(1)$-sized set of tiles. Single ticks between tiles represent a strength-1 glue, double ticks represent strength-2 glues. (b) Example Turing machine simulation. (c) The form of a computation assembly. (Color figure online)

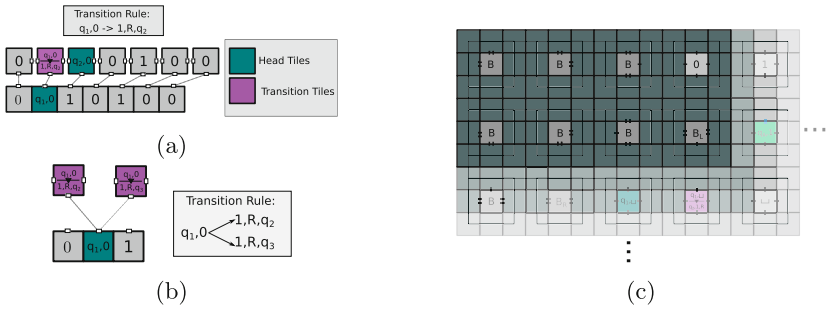


Fig. 8. (a) One step in the simulation of a Turing machine. The bottom row represents an initial valid Turing machine configuration. The tiles that can attach above this row represent a valid transition to another valid configuration of the Turing machine. (b) Simulation of nondeterministic transition rules. The glues on the south side of both of the transition tiles (purple) are the same, allowing either to be placed above the head tile (green). (c) Example target assembly. Solid lines represent unique strength-4 glues. Every tile type used in the TM simulation section and binary counter section can attach in only one spot. (Color figure online)

Turing Machine Simulation. We also use known techniques for simulating Turing machines in a self-assembly system [26]. An example of simulating one step is shown in Fig. 8a. We use this method to simulate the computation paths of M. Due to the nondeterministic nature of the model, we simulate nondeterministic transition rules by simply having a different tile type for each possible transition (Fig. 8b). For the instance $(M, k = 4)$, an example assembly (not stable) that represents an accepting computation path of M is shown in Fig. 7b. The system created by the reduction also includes the tile set necessary to simulate M in this manner. The set of tiles is disjoint from those used for the binary counter. In the same way, this tile set has the inner tiles (light gray) that perform the computation. These have a strength-1 glue on each side, and border tiles (dark gray). The north border uses a constant number of distinct tile types to ensure that the accept state of the Turing machine must be present in the row below it in order for a stable assembly to be formed.

Production of Computation Assembly. The key question of this system is whether the 70 hands can be utilized to bring together ≤ 70 of the described tiles to produce a computation assembly. In the case where the original instance K-A_{NTM} is true, the answer is 'yes'. For the example provided, 28 hands can be used to arrange the tiles of the binary counter section to form an assembly representing the counting from 0 to 4 (Fig. 7b). Above this, the remaining 42 hands can arrange up to 42 tiles in the Turing machine simulation section in an arrangement that represents an accepting computation path of at most 4 steps (Fig. 7b). The arrangement of binary counter tiles and Turing machine simulation tiles can form a stable assembly if attached to each other (Fig. 7c). Therefore the computation assembly is a producible assembly in a 70-handed system. Note that a

computation assembly of size less than 70 can also be produced if there exists an existing computation path strictly less than length 4.

Target Assembly. The target assembly for the H-HAM-UAV instance is an assembly that acts as a "frame" that holds all the tiles previously described (Fig. 8c). Each tile has a designated spot within this assembly that is specified by the frame having the corresponding glues that uniquely identify the tile adjacent to its designated position. Some consideration must be taken in the arrangement of the tiles within this frame to ensure that some extraneous assembly is not built within the frame. Since the tiles that compose the frame have strength-4 glues between them, it is clear that this frame is always built. For every tile in the binary counter section and Turing machine simulation section, there is one spot in the frame that exactly complements its glues, so it is true that each of these tiles can attach to the frame. Thus, the target assembly will always be produced.

5.3 Complexity

Given the membership and reduction overview in Sects. 5.1 and 5.2, respectively, we show the following.

Theorem 4. H-HAM-UAV *is complete for coNEXP.*

Proof. Lemma 5 shows that H-HAM-UAV is in the class coNEXP. The reduction shown is a polynomial time reduction from K-A$_{NTM}$ to H-HAM-UAV taking an instance $P = (M, k)$ to an instance $P' = (\Gamma = (T, h, 4), A)$ where $h = (2^{\lceil \log_2(k) \rceil} + 3) \cdot (k + \lceil \log_2(k) \rceil + 4)$, and $\neg P \iff P'$. It was shown how a true instance of P implies P' is false, through the production of a computation assembly that will never grow into the target assembly.

We now show that the instance P being false implies that the created instance of P' is true. It is clear the target assembly will be produced, but it must be shown that no assembly that is not a subassembly of the target assembly is produced. We first note that the border tiles can not come together alone to form a hollow square. This is because at the points where the border tiles from the binary counter section would meet those from the Turing machine simulation section, there are strength-1 glues (red arrows in Figs. 7a and 7b), meaning the hollow square is not a stable assembly.

From the provided tile types shown, in order to be stable the computation assembly must be enclosed by border tiles (dark gray). Every other tile has only strength-1 glues on every edge, and therefore if the tile were on the border, the assembly would not be stable. Due to a unique glue (shown in green in Fig. 7a), the right border of the binary counter can only be built if there is a 1 to the left of it in the row representing the most significant bit. Therefore, in order to be stable, the binary counter assembly must have counted up to $2^{\lceil \log_2(k) \rceil}$. Thus, $(2^{\lceil \log_2(k) \rceil} + 3) \cdot (\lceil \log_2(k) \rceil + 2)$ of the allotted hands must be used to "hold" the binary counter in place (28 in our example).

This leaves the system with $(2^{\lceil \log_2(k) \rceil} + 3) \cdot (k + 2)$ hands left, which can be used to arrange the Turing machine simulation tiles in a way that can attach to

the binary counter. Since only the border tiles of the binary counter assembly attach to the border tiles of the simulation assembly, the simulation assembly must be of the same width $(2^{\lceil \log_2(k) \rceil} + 3)$. Thus, it can be at most height $k + 2$. Since one row has to be used for the top border, the simulation can only utilize $k - 1$ rows, i.e., k steps. Due to another unique glue on the top border of the simulation assembly (cyan in Fig. 7b), the tile B_A can only be stable on an assembly if the row below it contains a tile that represents the accept state. Every tile in the Turing machine simulation section must have a matching glue with all of its neighbors. Since every two adjacent rows in the Turing machine simulation share matching glues, the glue encoding enforces that it is a valid transition from one configuration of the Turing machine to another. Therefore, starting from the initial configuration of M, if there does not exist a computation path that accepts in $\leq k$ steps, then there is no way to arrange the $k + 1$ rows in a way that is both stable and has the accept state present. Thus, if the instance P is false, then the only terminal assembly of the created system is the target assembly, so P' is true. □

Corollary 2. U-H-HAM-UAV *is complete for coNP.*

Proof. Corollary 1 shows that U-H-HAM-UAV is in the class coNP. The problem U-K-A_{NTM} where the parameter k denoting the maximum allotted runtime is encoded in unary is coNP-hard. An equivalent reduction which outputs the same instance $P' = (\Gamma = (T, h, 4), A)$ with the difference that h is encoded in unary is a polynomial time reduction from U-K-A_{NTM} to U-H-HAM-UAV. □

6 Conclusion

In this paper, we analyzed for the first time two of the most fundamental self-assembly questions in relation to the h-handed model: producibility and unique assembly verification. We proved that producibility is polynomial and UAV is coNP-complete when the number of hands is encoded in unary and coNEXP-complete if it is encoded in binary. Further, we gave a class of shapes that show the power of additional hands by having a provable separation in necessary tile types between shapes. We also showed that with a constant number of tile types, different sized squares are producible depending on the number of hands.

References

1. Adleman, L.M., et al.: Combinatorial optimization problems in self-assembly. In: Proceedings of the 34th Annual ACM Symposium on Theory of Computing, pp. 23–32 (2002)
2. Aggarwal, G., Cheng, Q., Goldwasser, M.H., Kao, M.Y., de Espanes, P.M., Schweller, R.T.: Complexities for generalized models of self-assembly. SIAM J. Comput. **34**(6), 1493–1515 (2005). https://doi.org/10.1137/S0097539704445202
3. Alon, N., Yuster, R., Zwick, U.: Finding and counting given length cycles. Algorithmica **17**, 209–223 (1997). https://doi.org/10.1007/BF02523189

4. Becker, F., Rapaport, I., Rémila, É.: Self-assemblying classes of shapes with a minimum number of tiles, and in optimal time. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 45–56. Springer, Heidelberg (2006). https://doi.org/10.1007/11944836_7
5. Bryans, N., Chiniforooshan, E., Doty, D., Kari, L., Seki, S.: The power of nondeterminism in self-assembly. In: Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 590–602. SIAM (2011)
6. Cannon, S., et al.: Two hands are better than one (up to constant factors): self-assembly in the 2HAM vs. aTAM. In: 30th International Symposium on Theoretical Aspects of Computer Science (STACS 2013). Leibniz International Proceedings in Informatics (LIPIcs), vol. 20, pp. 172–184. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2013)
7. Chalk, C., Luchsinger, A., Schweller, R., Wylie, T.: Self-assembly of any shape with constant tile types using high temperature. In: Proceedings of the 26th Annual European Symposium on Algorithms, ESA 2018 (2018)
8. Chalk, C.T., Fernandez, D.A., Huerta, A., Maldonado, M.A., Schweller, R.T., Sweet, L.: Strict self-assembly of fractals using multiple hands. Algorithmica **76**(1), 195–224 (2016). https://doi.org/10.1007/s00453-015-0022-x
9. Demaine, E.D., et al.: Staged self-assembly: nanomanufacture of arbitrary shapes with $O(1)$ glues. Nat. Comput. **7**(3), 347–370 (2008). https://doi.org/10.1007/s11047-008-9073-0
10. Doty, D.: Randomized self-assembly for exact shapes. SIAM J. Comput. **39**(8), 3521–3552 (2010)
11. Doty, D.: Theory of algorithmic self-assembly. Commun. ACM **55**(12), 78–88 (2012)
12. Doty, D.: Producibility in hierarchical self-assembly. In: Ibarra, O.H., Kari, L., Kopecki, S. (eds.) UCNC 2014. LNCS, vol. 8553, pp. 142–154. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08123-6_12
13. Eppstein, D.: Subgraph isomorphism in planar graphs and related problems. J. Graph Algorithms Appl. **3**(3), 1–27 (1999). https://doi.org/10.7155/jgaa.00014
14. Fekete, S.P., Schweller, R.T., Winslow, A.: Size-dependent tile self-assembly: constant-height rectangles and stability. In: Elbassioni, K., Makino, K. (eds.) ISAAC 2015. LNCS, vol. 9472, pp. 296–306. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48971-0_26
15. Kao, M.Y., Schweller, R.: Reducing tile complexity for self-assembly through temperature programming. arXiv preprint cs/0602010 (2006)
16. Kao, M.-Y., Schweller, R.: Randomized self-assembly for approximate shapes. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008. LNCS, vol. 5125, pp. 370–384. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70575-8_31
17. Kowalik, Ł: Short cycles in planar graphs. In: Bodlaender, H.L. (ed.) WG 2003. LNCS, vol. 2880, pp. 284–296. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39890-5_25
18. Kundeti, V., Rajasekaran, S.: Self assembly of rectangular shapes on concentration programming and probabilistic tile assembly models. Nat. Comput. **11**(2), 199–207 (2012). https://doi.org/10.1007/s11047-012-9313-1
19. Minev, D., Wintersinger, C.M., Ershova, A., Shih, W.M.: Robust nucleation control via crisscross polymerization of highly coordinated DNA slats. Nat. Commun. **12**(1), 1–9 (2021)

20. Patitz, M.J.: An introduction to tile-based self-assembly and a survey of recent results. Nat. Comput. **13**(2), 195–224 (2013). https://doi.org/10.1007/s11047-013-9379-4
21. Rothemund, P.W., Winfree, E.: The program-size complexity of self-assembled squares. In: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, pp. 459–468 (2000)
22. Schweller, R., Winslow, A., Wylie, T.: Complexities for high-temperature two-handed tile self-assembly. In: Brijder, R., Qian, L. (eds.) DNA 2017. LNCS, vol. 10467, pp. 98–109. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66799-7_7
23. Schweller, R., Winslow, A., Wylie, T.: Nearly constant tile complexity for any shape in two-handed tile assembly. Algorithmica **81**(8), 3114–3135 (2019). https://doi.org/10.1007/s00453-019-00573-w
24. Schweller, R., Winslow, A., Wylie, T.: Verification in staged tile self-assembly. Nat. Comput. **18**(1), 107–117 (2018). https://doi.org/10.1007/s11047-018-9701-2
25. Summers, S.M.: Reducing tile complexity for the self-assembly of scaled shapes through temperature programming. Algorithmica **63**(1), 117–136 (2012). https://doi.org/10.1007/s00453-011-9522-5
26. Winfree, E.: Algorithmic self-assembly of DNA. Ph.D. thesis, California Institute of Technology, June 1998
27. Woods, D.: Intrinsic universality and the computational power of self-assembly. Philos. Trans. Roy. Soc. A Math. Phys. Eng. Sci. **373**(2046), 20140214 (2015)
28. Woods, D., et al.: Diverse and robust molecular algorithms using reprogrammable DNA self-assembly. Nature **567**(7748), 366–372 (2019). https://doi.org/10.1038/s41586-019-1014-9