Centralised Connectivity-Preserving Transformations by Rotation: 3 Musketeers for all Orthogonal Convex Shapes

Matthew Connor and Othon Michail

Department of Computer Science, University of Liverpool, United Kingdom
`M.Connor3@liverpool.ac.uk, Othon.Michail@liverpool.ac.uk`

Abstract. We study a model of programmable matter systems consisting of n devices lying on the cells of a 2-dimensional square grid, which are able to perform the minimal mechanical operation of rotating around each other. The goal is to transform an initial shape of devices A into a target shape B. We are interested in characterising the class of shapes which can be transformed into each other in such a scenario, under the additional constraint of maintaining global connectivity at all times. This was one of the main problems left open by [Michail *et al.*, JCSS'19]. Note that the considered question is about structural feasibility of transformations, which we exclusively deal with via centralised constructive proofs. Distributed solutions are left for future work and form an interesting research direction. Past work made some progress for the special class of colour-consistent *nice* shapes. We here consider the class of *orthogonal convex shapes*, where for any two nodes u, v in a horizontal or vertical line on the grid, there is no empty cell between u and v. We develop a generic centralised transformation and prove that, for any pair A, B of colour-consistent orthogonal convex shapes, it can transform A into B. In light of the existence of *blocked* shapes in the considered class, we use a minimal 3-node *seed* (additional nodes placed at the start) to trigger the transformation. The running time of our transformation is an optimal $O(n^2)$ sequential moves, where $n = |A| = |B|$. We leave as an open problem the existence of a universal connectivity-preserving transformation with a small seed. Our belief is that the techniques developed in this paper might prove useful to answer this.

Keywords: Programmable matter · Transformation · Reconfigurable robotics · Shape formation · Centralised algorithms.

The full paper including all omitted details is available on arXiv at: https://arxiv.org/abs/2207.03062

1 Introduction

Programmable matter refers to matter which can change its physical properties algorithmically. This means that the change is the result following the procedure of an underlying program. The implementation of the program can either be a system-level external centralised algorithm or an internal decentralised algorithm executed by the material itself. The model for such systems can be further refined to specify properties that are relevant to real-world applications, for example connectivity, colour [4] and other physical properties.

As the development of these systems continues, it becomes increasingly necessary to develop theoretical models which are capable of describing and explaining the emergent properties, possibilities and limitations of such systems in an abstract and fundamental manner. To this end, models have been developed for programmable matter. For example, algorithmic self-assembly [10,19] focuses on programming molecules like DNA to grow in a controllable way, and the Abstract Tile Assembly Model [20,25], the Kilobot model [21], the Robot Pebbles system [14], and the nubot model [26], have all been developed for this area. Network Constructors [18] is an extension of population protocols [3] that allows for network formation and reconfiguration. The latter model is formally equivalent to a restricted version of chemical reaction networks, which "are widely used to describe information processing occurring in natural cellular regulatory networks" [22,11]. The CATOMS system [23,24,12] is a further implementation which constructs 3D shapes by first creating a "scaffolding structure" as a basis for construction. Finally, there is extensive research into the amoebot model [7,6,9,8], where finite automata on a triangular lattice follow a distributed algorithm to achieve a desired goal, including a recent extension [13] to a circuit-based model.

Recent progress in this direction has been made in previous papers, for example [17], covering questions related to a specific model of programmable matter where nodes exist in the form of a shape on a 2D grid and are capable of performing two specific movements: rotation around each other and sliding a node across two other nodes. The authors investigated the problem of transformations with rotations with the restriction that shapes must always remain connected (RotC-Transformability), and left universal RotC-Transformability as an open problem. They hinted at the possibility of universal transformation in an arxiv draft [16]. To the best of our knowledge, progress on this open question has only been made in [5], where, by using a small seed, connectivity-preserving transformations by rotation were developed for a restricted class of shapes. In general, such transformations are highly desirable due to the large numbers of programmable matter systems which rely on the preservation of connectivity and the simplicity of movement, which is not only of theoretical interest but is also more likely to be applicable to real-world systems. Related progress was also made in [1], which used a similar model but with a different type of movement. The authors allowed for a greater range of movement, for example "leapfrog" and "monkey" movements. They accomplished universal transformation in $O(n^2)$ movements using a "bridging" procedure assisted by at most 5 seed-nodes, which they called *musketeers*.

2 Contribution

We investigate the RotC-Transformability problem, introduced in [17], which asks to characterise which families of connected shapes can be transformed into each other via rotation movements without breaking connectivity. The model represents programmable matter on a 2D grid which is only capable of performing rotation movements, defined as the 90° rotation of a node u around a neighbouring edge-adjacent node v, so long as the goal and intermediate cells are empty. As our focus is on the feasibility and complexity of transformations, our approach is naturally based on structural characterisations and *centralised* procedures. Structural and algorithmic progress is expected to facilitate more applied future developments, such as distributed implementations.

We assume the existence of a *seed*, a group of nodes in a shape S which are placed in empty cells neighbouring a shape A to create a new connected shape which is the unification of S and A. Seeds allow shapes which are blocked or incapable of meaningful movement to perform otherwise impossible transformations. The use of seeds was established in [17], leaving open the problem of universal RotC-Transformability. Another work [5] investigated this problem in the context of *nice* shapes, first defined in [2] as a set of shapes containing any shape S which has a central line L, where, for all nodes $u \in S$, either $u \in L$ or u is connected to L by a line of nodes perpendicular to L. Universal reconfiguration in the context of connectivity-preserving transformations using different types of movement has been demonstrated in [1]. That paper calls the seed nodes "musketeers" and their transformation requires the use of 5 such nodes.

The present paper moves towards a solution which is based on connectivity-preservation and the tighter constraints of rotation-only movement of [17] while aiming to (i) widen the characterization of the class of transformable shapes and (ii) minimise the seed required to trigger those transformations. By achieving these objectives for orthogonal convex shapes, we make further progress towards the ultimate goal of an exact characterisation (possibly universal) for seed-assisted RotC-Transformability.

We study the transformation of shapes of size n with *orthogonal convexity* into other shapes of size n with the same property, via the canonical shape of a *diagonal line-with-leaves*. Orthogonal convexity is the property that for any two nodes u, v in a horizontal or vertical line on the grid, there is no empty cell between u and v. A diagonal line-with-leaves is a group of components, the main being a series of 2-node columns where each column is offset such that the order of the nodes is equivalent to a line, and two optional components: two 1-node columns on either end of the shape and additional nodes above each column, making them into 3-node columns. We show that transforming a orthogonal convex shape of n nodes into a diagonal line-with-leaves is possible and can be achieved by $O(n^2)$ moves using a 3-node seed. This bound on the number of moves is optimal for the considered class, due to a matching lower bound from [17] on the distance between a line and a staircase, both of which are orthogonal convex shapes. A seed is necessary due to the existence of blocked orthogonal convex shapes, an example being a rhombus. As [5] shows, any seed with less than 3 nodes is incapable of non-trivial transformation of a line of nodes. Since a line of nodes is orthogonal convex, the 3-node seed employed here is minimal.

The class of orthogonal convex shapes cannot easily be compared to the class of nice shapes. A diagonal line of nodes in the form of a staircase belongs to the former but not the latter. Any nice shape containing a gap between two of its columns is not a orthogonal convex shape. Finally, there are shapes like a square of nodes which belong to both classes. Nevertheless, the nice shapes that are not orthogonal convex have turned out to be much easier to handle than the orthogonal convex shapes that are not nice. We hope that the methods we had to develop in order to deal with the latter class of shapes, will bring us one step closer to an exact characterisation of connectivity-preserving transformations by rotation.

In Section 3, we formally define the programmable matter model used in this paper. Section 4 presents some basic properties of orthogonal convex shapes and of their elimination and generation sequences. In Section 5, we provide our algorithm for the construction of the diagonal line-with-leaves which, through reversibility, can be used to construct other orthogonal convex shapes and give time bounds for it. In Section 6, we conclude and give directions for potential future research.

3 Model

We consider the case of programmable matter on a 2D grid, with each position (or cell) of the grid being uniquely referred to by its (x, y) coordinates. Such a system consists of a set V of n nodes. Each node may be viewed as a spherical module fitting inside a cell of the grid. At any given time, each node occupies a cell, with the positioning of the nodes defining a shape, and no two nodes may occupy the same cell. It also defines an undirected *neighbouring relation* $E \subset V \times V$, where $uv \in E$ iff u and v occupy *horizontally* or *vertically* adjacent cells of the grid. A shape is *connected* if the graph induced by its neighbouring relation is a connected graph.

In general, shapes can *transform* to other shapes via a sequence of one or more movements of individual nodes. We consider only one type of movement: *rotation*. In this movement, a single node moves relative to one or more neighbouring nodes. A single rotation movement of a node u is a $90°$ rotation of u around one of its neighbours. Let (x, y) be the current position of u and let its neighbour be v occupying the cell $(x, y - 1)$ (i.e., lying below u). Then u can rotate $90°$ clockwise (counterclockwise) around v iff the cells $(x + 1, y)$ and $(x + 1, y - 1)$ ($(x - 1, y)$ and $(x - 1, y - 1)$, respectively) are both empty. By rotating the whole system $90°$, $180°$, and $270°$, all possible rotation movements can be defined.

Let A and B be two connected shapes. We say that A transforms to B via a rotation r, denoted $A \xrightarrow{r} B$, if there is a node u in A such that if u applies r, then the shape resulting after the rotation is B. We say that A transforms in one step to B (or that B is reachable in one step from A), denoted $A \to B$, if $A \xrightarrow{r} B$ for some rotation r. We say that A transforms to B (or that B is reachable from

A) if there is a sequence of shapes $A = S_1, S_2, \dots, S_t = B$, such that $S_i \rightarrow S_{i+1}$ for all $1 \leq i \leq t-1$. Rotation is a reversible movement, a fact that we use in our results. All shapes $S_1, S_2, \dots, S_t$ must be *edge connected*, meaning that the graph defined by the neighbouring relation E of all nodes in any S_i, where $1 \leq i \leq t$, must be a connected graph.

At the start of the transformation, we will be assuming the existence of a *seed*: a small connected shape M placed on the perimeter of the given shape S to trigger the transformation. This is essential because under rotation-only there are shapes S that are k-blocked, meaning that at most $k \geq 0$ moves can be made before a configuration is repeated. When $k = 0$, no move is possible from S, an example of 0-blocked shape being the rhombus.

For the sake of providing clarity to our transformations, we say that every cell in the 2D grid has a colour from $\{red, black\}$ in such a way that the cells form a black and red checkered colouring of the grid, similar to the colouring of a chessboard. This colouring is fixed so long as there is at least one node on the grid. This represents a property of the rotation movement, which is that any given node in a coloured cell can only enter cells of the same colour. We define $c(u) \in \{black, red\}$ as the colour of node u for a given chessboard colouring of the grid. We represent this in our figures by colouring the nodes red or black. See Figure 1 for an example and for special notation that we use to abbreviate certain rotations which we perform throughout the paper.

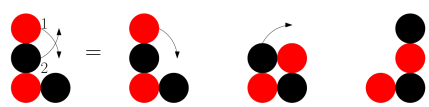


Fig. 1: The rotation on the left is an abbreviated version of the rotations on the right, used throughout the paper. The numbers represent the order of rotations. Red nodes appear grey in print, throughout the paper.

Any shape S may be viewed as a coloured shape consisting of $b(S)$ blacks and $r(S)$ reds. Two shapes A and B are *colour-consistent* if $b(A) = b(B)$ and $r(A) = r(B)$. For any shape S of n nodes, the *parity* of S is the colour of the majority of nodes in S. If there is no strict majority, we pick any as the parity colour. We use *non-parity* to refer to the colour which is not the parity.

Depending on the context and purpose, the term *node* will be used to refer both to the actual entity that may move between co-ordinates and to the co-ordinates of that entity at a given time.

The *perimeter* of a connected shape S is the minimum-area polygon that completely encloses S in its interior, existence of an interior and exterior directly following from the Jordan curve theorem [15]. The *cell perimeter* of S consists of every cell of the grid not occupied by S that contributes at least one of its edges to the perimeter of S. The *external surface* of S consists of all nodes $u \in S$ such that u occupies a cell defining at least one of the edges of the perimeter of

S. The *extended external surface* of S is then defined by adding to the external surface all nodes of S whose cell shares a corner with the perimeter of S. A *line* (of nodes) of length k is a series of consecutive nodes $u_1, u_2, \ldots, u_k$ in a given row or column. For a line $u_1, u_2, \ldots, u_k$, we refer to u_1 and u_k as the *end nodes* or *endpoints* of the line. Our proofs make use of column and row analysis, by dividing connected orthogonal convex shapes into p rows $R_1, R_2, \ldots, R_p$ and q columns $C_1, C_2, \ldots, C_q$. We assume without loss of generality (abbreviated to "w.l.o.g." throughout) that R_1 and C_1 are the bottom-most row and leftmost column, respectively. We use $a \times b$ to refer to a rectangle of a rows and b columns, where all rows and columns are fully occupied.

We use σ and variants to denote sequences of nodes. A *k-sub-sequence* σ' of a sequence σ is any sub-sequence of σ where $|\sigma'| = k$. For a given colouring of the grid, the *colour sequence* $c(\sigma)$ of a sequence of nodes $\sigma = (u_1, u_2, \ldots, u_n)$ is defined as $c(\sigma) = (c(u_1), c(u_2), \ldots, c(u_n))$. A sequence σ' is *colour-order preserving w.r.t* σ if $c(\sigma') = c(\sigma)$. A sequence of nodes $\sigma = (u_1, u_2, \ldots, u_n)$ is called *single-coloured* if $c(\sigma)$ satisfies $c(u_i) = s$ for all $1 \leq i \leq n$ and some $s \in \{black, red\}$.

4 Preliminaries

4.1 Orthogonal Convex Shapes

We now present the class of shapes considered in this paper together with some basic properties about them that will be useful later.

Definition 1. *A shape S is said to belong to the family of* orthogonal convex *shapes, if, for any pair of distinct nodes $(x_1, y_1), (x_2, y_2) \in S$, $x_1 = x_2$ implies $(x_1, y) \in S$ for all $\min\{y_1, y_2\} < y < \max\{y_1, y_2\}$ while $y_1 = y_2$ implies $(x, y_1) \in S$ for all $\min\{x_1, x_2\} < x < \max\{x_1, x_2\}$.*

Observe now that the perimeter of any connected shape is a cycle drawn on the grid, i.e., a path where its end meets its beginning. The cycle is drawn by using consecutive grid-edges of unit length, each being characterized by a direction from $\{up, right, down, left\}$. For each pair of opposite directions, $(up, down)$ and $(left, right)$, the perimeter always uses an equal number of edges of each of the two directions in the pair and uses every direction at least once. For the purposes of the following proposition, let us denote $\{up, right, down, left\}$ by $\{d_1, d_2, d_3, d_4\}$, respectively. The perimeter of a shape can then be defined as a sequence of moves drawn from $\{d_1, d_2, d_3, d_4\}$, w.l.o.g. always starting with a d_1. Let also N_i denote the number of times d_i appears in a given perimeter.

Proposition 1. *A shape S is a connected orthogonal convex shape if and only if its perimeter satisfies both the following properties:*

- *It is described by the regular expression*

$$d_1(d_1 \mid d_2)^* d_2(d_2 \mid d_3)^* d_3(d_3 \mid d_4)^* d_4(d_4 \mid d_1)^*$$

under the additional constraint that $N_1 = N_3$ and $N_2 = N_4$.

– *Its interior has no empty cell.*

Proof. We begin by considering the forward direction, starting from a connected orthogonal convex shape S. For the first property, the N_i equalities hold for the perimeter of any shape, thus, also for the perimeter of S. In the regular expression, the only property that is different from the regular expressions of more general perimeters is that, for all $i \in \{1, 2, 3, 4\}$, d_{i-2}, where the index is modulo 4, does not appear between the first and the last appearance of d_i.

Assume that it does, for some i. Then d_{i-2} must have appeared immediately after a d_{i-1} or a d_{i+1}, because a d_{i-2} can never immediately follow a d_i. If it is after a d_{i-1}, then this forms the expression $d_i(d_{i-1} \mid d_i)^* d_{i-1}d_{i-2}$, which always has $d_i d_{i-1}^+ d_{i-2}$ as a sub-expression. But for any sub-path of the perimeter defined by the latter expression, the nodes attached to its first and last edges would then contradict Definition 1, as the horizontal or vertical line joining them goes through at least one unoccupied cell, i.e., one of the cells external to the d_{i-1}^+ part of the sub-path. The d_{i+1} case follows by observing that, in this case, the sub-expression satisfied by the perimeter would be $d_{i-2}d_{i+1}^+ d_i$, which would again violate the orthogonal convexity of S.

The second property, follows immediately by observing that if (x, y) is an empty cell within the perimeter's interior, then the horizontal line that goes through (x, y) must intersect the perimeter at two distinct points, one to the left of (x, y) and one to its right. Thus, these three points would contradict the conditions of Definition 1.

For the other direction, let S be a shape satisfying both properties. For the sake of contradiction, assume that S is not orthogonal convex, which means that there is a line, w.l.o.g horizontal and of the form $(x_l, y), (x_l + 1, y), \ldots, (x_r, y)$, where (x_l, y) and (x_r, y) are occupied by nodes of S while $(x_l+1, y), \ldots, (x_r-1, y)$ are not. Observe first that any gap in the interior would violate the second property, thus $(x_l + 1, y), \ldots, (x_r - 1, y)$ must be cells in the exterior of the perimeter of S and (x_l, y), (x_r, y) nodes on the perimeter. There are two possible ways to achieve this: $d_3d_2^+d_1$ and $d_1d_4^+d_3$. These combinations are impossible to create with the regular expression, thus contradicting that S satisfies the properties. Similarly for vertical gaps. It follows that any shape fulfilling the two properties must belong to the family of connected orthogonal convex shapes. □

Let c_x denote the column of a given shape S at the x coordinate, i.e., the set of all nodes of S at x. Let $y_{max}(x)$ ($y_{min}(x)$) be the largest (smallest) y value in the (x, y) coordinates of the cells which nodes of a column c_x occupy.

Proposition 2. *For any connected orthogonal convex shape S, all the following are true:*

– *Every column c_x of S consists of the consecutive nodes* $(x, y_{min}(x)), (x, y_{min}(x) + 1), \ldots, (x, y_{max}(x))$.
– *There are no three columns c_{x_1}, c_{x_2}, and c_{x_3} of S, $x_1 < x_2 < x_3$, for which both $y_{max}(x_1) > y_{max}(x_2)$ and $y_{max}(x_3) > y_{max}(x_2)$ hold.*

– *There are no three columns $c_{x'_1}$, $c_{x'_2}$, and $c_{x'_3}$ of S, $x'_1 < x'_2 < x'_3$, for which both $y_{min}(x'_1) < y_{min}(x'_2)$ and $y_{min}(x'_3) < y_{min}(x'_2)$ hold.*

All the above hold for rows too in an analogous way.

Lemma 1. *For all $n \geq 3$, the maximum colour-difference of a connected horizontal-vertical convex shape of size n is $n - 2\lfloor n/3 \rfloor$.*

A *staircase* is a shape of the form $(x, y), (x+1, y), (x+1, y+1), (x+2, y+1), \ldots$ or $(x, y), (x, y+1), (x+1, y+1), (x+1, y+2), \ldots$. An *extended staircase* is a staircase $Stairs = \{(x_l, y_d), (x_l, y_d+1), (x_l+1, y_d+1), (x_l+1, y_d+2), \ldots\}$ with a bicolour pair at $(x_l - 1, y_d)$, $(x_l - 1, y_d + 1)$ or at $(x_l - 1, y_d - 1)$, $(x_l - 1, y_d)$. Additionally, there are three optional *node-repositories*, *BRep*, *RRep* and a single-black repository. $BRep = \{(x_l, y_d + 2), (x_l + 1, y_d + 3), (x_l + 2, y_d + 4), \ldots\}$, $RRep = \{(x_l, y_d - 1), (x_l + 1, y_d), (x_l + 2, y_d + 1), \ldots\}$ and the single-black repository at $(x_l - 2, y_d)$.

4.2 Elimination and Generation Sequences

Let S be a connected orthogonal convex shape. A *shape elimination sequence* $\sigma = (u_1, u_2, \ldots, u_n)$ of S defines a sequence $S = S_0[u_1]S_1[u_2]S_2[u_3] \ldots S_{n-1}[u_n]S_n = \emptyset$, where, for all $1 \leq t \leq n$, a connected orthogonal convex shape S_t is obtained by removing the node u_t from the external surface of the shape S_{t-1}. A *row elimination sequence* σ of S is an elimination sequence of S which consists of p sub-sequences $\sigma = \sigma_1\sigma_2 \ldots \sigma_p$, each sub-sequence σ_i, $1 \leq i \leq p$, satisfying the following properties. Sub-sequence σ_i consist of the $k = |R_i|$ nodes of row R_i, where $u_1, u_2, \ldots, u_k$ is the line formed by row R_i. Additionally, σ_i is of the form $\sigma_i = \sigma_i^1\sigma_i^2$, where (i) $\sigma_i^1 = (u_1, \ldots, u_k)$ or $\sigma_i^1 = (u_k, \ldots, u_1)$ and σ_i^2 is empty or (ii) there is a $u_j \in R_i$, for $2 \leq j < k$, such that $\sigma_i^1 = (u_1, \ldots, u_j)$ and $\sigma_i^2 = (u_k, \ldots, u_{j+1})$ or (iii) there is a $u_j \in R_i$, for $1 \leq j < k - 1$, such that $\sigma_i^1 = (u_k, \ldots, u_{j+2})$ and $\sigma_i^2 = (u_1, \ldots, u_{j+1})$. We shall call any such sub-sequence σ_i an elimination sequence of row R_i.

Given a connected orthogonal convex shape S of n nodes, a *shape generation sequence* $\sigma = (u_1, u_2, \ldots, u_n)$ of S defines a sequence $\emptyset = S_0[u_1]S_1[u_2]S_2[u_3] \ldots S_{n-1}[u_n]S_n = S$, where, for all $1 \leq t \leq n$, a connected orthogonal convex shape S_t is obtained by adding the node u_t to the cell perimeter of S_{t-1}.

Let S be an extended staircase of n nodes. An *extended staircase generation sequence* $\sigma = (u_1, u_2, \ldots, u_n)$ of S is a generation sequence of S which consists of q sub-sequences $\sigma = \sigma_1\sigma_2 \ldots \sigma_q$, where each σ_i contains the nodes of the column C_i of S, ordered such that they do not violate the properties of a shape generation sequence. A *diagonal line-with-leaves generation sequence* can be defined in an analogous way.

Lemma 2. *Every connected orthogonal convex shape S has a row (and column) elimination sequence σ.*

Lemma 3. *Let σ be a bicoloured sequence of nodes that fulfills all the following conditions:*

- *The set of the first two nodes in* σ *is not single-coloured.*
- *The third node of* σ *is black.*
- σ *does not contain a single-coloured 3-sub-sequence.*

Then there is an extended staircase generation sequence $\sigma' = (u'_1, u'_2, \ldots, u'_n)$ *which is colour-order preserving with respect to* σ*.*

Lemma 4. *For any connected orthogonal convex shape* S *of* n *nodes, given a row elimination sequence* $\sigma = (u_1, u_2, \ldots, u_n)$ *of* S *where the set of the first two nodes in* σ *is not single-coloured and* u_3 *is black, there is an extended staircase generation sequence* $\sigma' = (u'_1, u'_2, \ldots, u'_n)$ *which is colour-order preserving w.r.t* σ *and such that, for all* $1 \leq i \leq |\sigma|$, $D_i = \{u'_1, u'_2, \ldots, u'_i\}$ *is a connected orthogonal convex shape.*

The *anchor node* of the shape S of p rows $R_1, R_2, \ldots, R_p$ is the rightmost node in the row R_p, counting rows from bottom to top. *ExtendedStaircase* is an algorithm which creates an extended staircase generation sequence from a row elimination sequence of a connected horizontal-vertical convex shape.

Lemma 5. *Let* S *be a connected orthogonal convex shape of* n *nodes divided into* p *rows* $R_1, R_2, \ldots, R_p$*, and* $\sigma = (u_1, u_2, \ldots, u_n)$ *a row elimination sequence from* R_1 *to* R_p *of* S*. If the bottom node of the first two nodes placed by Extended-Staircase is fixed to* $(x_c, y_c + 1)$*, where* (x_c, y_c) *are the co-ordinates of the anchor node of* S*, the shape* $T_i = \mathit{ExtendedStaircase}(\sigma_i)$*, where* $\sigma_i = (u_1, u_2, \ldots, u_i)$*,* $1 \leq i \leq n$*, fulfills the following properties:*

- $S \cup T_i$ *is a connected shape.*
- $S \cap T = \emptyset$*.*
- *excluding the single-black repository,* $R_p \cup T_i$ *is an orthogonal convex shape.*

Lemma 6. *For any extended staircase* $W \cup T$ *of* n *nodes, where* W *is the Stairs,* $T \subseteq \{BRep \cup RRep\}$ *and* $k = |T|$*, given a shape elimination sequence* $\sigma = (u_1, u_2, \ldots, u_k)$ *of* T*, there is a diagonal line-with-leaves generation sequence* $\sigma' = (u'_1, u'_2, \ldots, u'_k)$ *which is colour-order preserving w.r.t* σ *and such that, for all* $1 \leq i \leq |\sigma|$*,* $D_i = W \cup \{u'_1, u'_2, \ldots, u'_i\}$ *is a connected orthogonal convex shape.*

5 The Transformation

In this section, we present the transformation of *orthogonal convex* shapes, via an algorithm (Algorithm 1) for constructing a diagonal line-with-leaves from any orthogonal convex shape S. For the first step of the algorithm, we generate a 6-robot from the seed and the shape, which we then use to transport nodes. By using a row elimination sequence of S and an extended staircase generation sequence, we convert the initial shape S into an extended staircase. We then use appropriate elimination and generation sequences focused on the repositories of the extended staircase, to convert the latter into a diagonal line-with-leaves.

Given any two colour-consistent orthogonal convex shapes A and B and their diagonal line-with-leaves D, our algorithm can be used to transform both A into D and B into D and, thus, A into B, by reversing the latter transformation. This transformation applies to all orthogonal convex shapes with 3 nodes. A 2-node shape can trivially transform by rotating one node around the other and a 1-node shape cannot transform at all.

Algorithm 1 OConvexToDLL(S, M)

Input: shape $S \cup M$, where S is a connected orthogonal convex shape of n nodes and M is a 3-node seed on the cell perimeter of S, row elimination sequence $\sigma = (u_1, u_2, \dots, u_n)$ of S, extended staircase generation sequence of $W \cup T = \sigma' = (u'_1, u'_2, \dots, u'_n)$ which is colour-order preserving w.r.t. σ, shape elimination sequence $\sigma = (u_1, u_2, \dots, u_{|T|})$ of T, shape generation sequence of $X = \sigma' = (u'_1, u'_2, \dots, u'_{|T|})$ which is colour preserving w.r.t. σ
Output: shape $G = W \cup X \cup M$, where G is a diagonal line-with-leaves and M is a connected 3-node shape on the cell perimeter of S.

```
R ← GenerateRobot(S, M)
σ ← rowEliminationSequence(S)
σ' ← ExtendedStaircase(σ)
W ∪ T ← HVConvexToExtStaircase(S, R, σ, σ')
σ ← repsEliminationSequence(W ∪ T)
σ' ← stairExtensionSequence(W ∪ T)
G ← ExtStaircaseToDLL(W ∪ T, R, σ, σ')
TerminateRobot(G, R)
```

Algorithm 2 OConvexToExtStaircase(S, R, σ, σ')

Input: shape $S \cup R$, where S is a connected orthogonal convex shape of n nodes and R is a 6-node robot on the cell perimeter of S, row elimination sequence $\sigma = (u_1, u_2, \dots, u_n)$ of S, extended staircase generation sequence $\sigma' = (u'_1, u'_2, \dots, u'_n)$ which is colour-order preserving w.r.t. σ
Output: shape $T \cup R$, where T is the extended staircase generated by σ'

```
for all 1 ≤ i ≤ n do
    source ← σ_i, dest ← σ'_i
    while R cannot extract source do
        if R can climb then Climb(R)
        else Slide(R)
    end while
    Extract(R, source)
    while R cannot place its load in dest do
        if R can climb then Climb(R)
        else Slide(R)
    end while
    Place(R, dest)
end for
```

Algorithm 3 ExtStaircaseToDLL(W, R, σ, σ')

Input: extended staircase $W = Stairs \cup \{BRep \cup RRep\}$ and a 6-robot R on its cell perimeter, shape elimination sequence $\sigma = (u_1, u_2, \ldots, u_{|T|})$ of $T \subseteq \{BRep \cup RRep\}$, shape generation sequence $\sigma' = (u'_1, u'_2, \ldots, u'_{|T|})$ which is colour-order preserving w.r.t. σ

Output: shape $Stairs' \cup R'$, where $Stairs' \setminus Stairs$ is an extension of $Stairs$ generated by σ' and R' is a 6-robot which is colour-consistent with R.

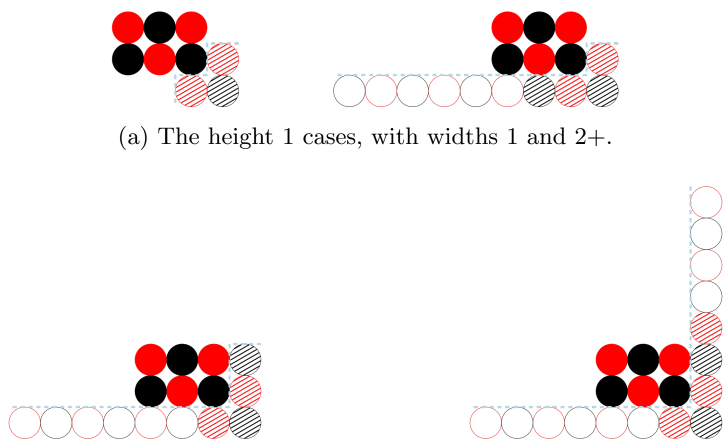
```
for all 1 ≤ i ≤ |T| do
    source ← u_i
    dest ← u'_i
    while R not at source do
        if R can climb then
            ClimbTowards(R, source)
        else
            SlideTowards(R, source)
        end if
    end while
    Extract(R, source)
    while R not at dest do
        if R can climb then
            ClimbTowards(R, dest)
        else
            SlideTowards(R, dest)
        end if
    end while
    Place(R, dest)
end for
```

5.1 Robot Traversal Capabilities

6-Robot Movement We first show that for all S in the family of orthogonal convex shapes, a connected 6-robot is capable of traversing the perimeter of S. We prove this by first providing a series of scenarios which we call *corners*, where we show that the 6-robot is capable of making progress past the obstacle that the corner represents. We then use Proposition 1 to show that the perimeter of any S is necessarily made up of a sequence of such corners, and therefore the 6-robot is capable of traversing it.

We begin by considering the *up-right* quadrant, that is any cells which neighbour the section of the perimeter defined by the regular expression $d_1(d_1 \mid d_2)^*d_2$ $(d_2 \mid d_3)^*d_3$, where d_1, d_2 and d_3 are *up*, *right* and *down* respectively, as our base case. We define progress as the movement of the 6-robot upwards and to the right of its starting position. Our goal is to show that attaining the maximum progress for each corner is possible. Since we can construct a series of corners where every corner follows from the point of maximum progress of the previous corner, it follows that for such a series we can make progress indefinitely.

Let $\mathcal{C}$ be a set of orthogonal convex shapes, where each shape is a corner scenario for the $up-right$ quadrant, depicted in Figure 2. Given a corner-shape scenario $C \in \mathcal{C}$ consisting of a horizontal line $(x_l, y_d), (x_l+1, y_d), \ldots, (x_r, y_d)$ and a vertical line $(x_r, y_d), (x_r, y_d+1), \ldots, (x_r, y_u)$, as depicted in Figure 2, we define its *width* $w(C) = |x_r - x_l|$, i.e., equal to the length of its horizontal line, and its *height* $h(C) = |y_u - y_d|$, i.e., equal to the length of its vertical line, excluding in both cases the corner node (x_r, y_d).



(a) The height 1 cases, with widths 1 and 2+.

(b) The height 2 case.

(c) The height 3+ case.

Fig. 2: The four basic corner scenarios of $\mathcal{C}$. Filled circles represent the 6-robot. Striped circles represent the nodes on the exterior of the shape. Hollow circles represent potential space for additional nodes for corner scenarios which are not in this set.

Lemma 7. *For any orthogonal convex shape S, the extended external surface defined by the regular expression $d_1(d_1 \mid d_2)^* d_2\ (d_2 \mid d_3)^* d_3$ of the shape can be divided into a series of shapes $S_0, S_1, \ldots,$ where all $S_i \in \mathcal{C}$.*

Given that the quadrant is made up of cases from $\mathcal{C}$, if the 6-robot is able to move from one vertical to another for all $S_i \in \mathcal{C}$, it is able to do so for any upright quadrant of the perimeter until it runs into the d_3 line. We now show that this movement is possible, first for this quadrant and later for all four quadrants.

Lemma 8. *For all shapes $C \in \mathcal{C}$, if a 2×3 shape (the 6-robot) is placed in the cells $(x_l-2, y_d+1), (x_l-1, y_d+1), (x_l, y_d+1), (x_l-2, y_d+2), (x_l-1, y_d+2), (x_l, y_d+2)$, it is capable of translating itself to $(x_r-2, y_u+1), (x_r-1, y_u+1), (x_r, y_u+1), (x_r-2, y_u+2), (x_r-1, y_u+2), (x_r, y_u+2)$.*

Theorem 1. *For any orthogonal convex shape S, a 6-robot is capable of traversing the perimeter of S.*

7-Robot Movement We consider once again the *up-right* quadrant, and generalise to other quadrants later. We say a cell $c = (x, y)$ is *behind* the robot if x is smaller than the x-coordinate of every node in the robot.

The *load* of a 7-robot S is any node u such that $S \setminus \{u\}$ is a 2×3 shape. The *position* of the robot is an offset of the y axis for the purpose of the initial positioning of the 7-robot. For our transformations, we maintain the invariant that the 7-robot, after any of its high-level movements, will return to the structure of a 2×3 shape with a load. For this invariant, we assume that the load is always *behind* the 2×3 shape (while remaining connected). The situation where the load is positioned differently does not need to be considered. We therefore use $(x, y|y')$ to refer to the co-ordinates of the two cells (x, y) and (x, y') *behind* the robot which can contain the load, keeping it attached to the robot while the latter is a 2×3 shape.

Lemma 9. *For all shapes $C \in \mathcal{C}$, if a 2×3 shape with a load (the 7-robot) is placed in the cells $(x_l - 3, y_d + 1|y_d + 2), (x_l - 2, y_d + 1), (x_l - 1, y_d + 1), (x_l, y_d + 1), (x_l - 2, y_d + 2), (x_l - 1, y_d + 2), (x_l, y_d + 2)$, it is capable of translating itself to $(x_r - 3, y_u + 1|y_u + 2), (x_r - 2, y_u + 1), (x_r - 1, y_u + 1), (x_r, y_u + 1), (x_r - 2, y_u + 2), (x_r - 1, y_u + 2), (x_r, y_u + 2)$.*

Theorem 2. *For any orthogonal convex shape S, a 7-robot is capable of traversing the perimeter of S.*

Repository Traversal Whenever the single-black repository is occupied, the robot may need to traverse a non-convex region when moving between S and the extended staircase. The following lemma shows that this is not an issue.

Lemma 10. *If the single-black repository of the extended staircase is occupied, then both the 6-robot and the 7-robot are able to traverse past it.*

5.2 Initialisation

Robot Generation We now prove that we can generate a 6-robot from the orthogonal convex shape S with the help of the 3 musketeers.

Lemma 11. *Let S be a connected orthogonal convex shape. Then there is a connected shape M of 3 nodes (the 3 musketeers) and an attachment of M to the bottom-most row of S, such that $S \cup M$ can reach a configuration $S' \cup M'$ satisfying the following properties. $S' = S \setminus \{u_1, u_2, u_3\}$, where $\{u_1, u_2, u_3\}$ is the 3-prefix of a row elimination sequence σ of S starting from the bottom-most row of S. M' is a 6-robot on the perimeter of S'.*

Prefix Construction To construct the extended staircase from an orthogonal convex shape S, we must first retrieve a sequence of 3 nodes u_1, u_2, u_3 from S, where u_3 is black. We assume w.l.o.g. that S is a black parity shape. We now show with the following lemma that this is possible. We consider the edge case where S is a rhombus in the full version of our paper.

Lemma 12. *For any shape $S \cup M$, where S is a non-red parity connected orthogonal convex shape of n nodes divided into p rows, $R_1, R_2, \ldots, R_p$ and M is a 6-robot, it is possible for M to extract a sequence of nodes (u_1, u_2, u_3) from S, where u_1, u_2 is a bicolour pair, u_3 is black, and $S \setminus \{u_1, u_2, u_3\}$ is a connected orthogonal convex shape.*

5.3 Transformations between Shapes

In this section, we show that, given our previous results, we are now in the position to convert an orthogonal convex shape into another such shape. We begin with the conversion of an extended staircase into a diagonal line-with-leaves, then the orthogonal convex shape to the diagonal line-with-leaves, and then our main result follows by reversibility.

Lemma 13. *Let S be a connected orthogonal convex shape with n nodes divided into p rows $R_1, R_2, \ldots, R_p$. Given a row elimination sequence $\sigma = (u_1, u_2, \ldots, u_n)$ of S, an extended staircase generation sequence $\sigma' = (u'_1, u'_2, \ldots, u'_n)$ which is colour-order preserving w.r.t. σ, and a 6-robot placed on the external surface of S, for all $1 \leq i < n$ the 6-robot is capable of picking up the node u_i, moving as a 7-robot to the empty cell u'_i and placing it, and then returning as a 6-robot to u_{i+1}.*

Proof. We follow the procedure of Algorithm 2. By Theorem 1 and Theorem 2, the 6-robot R and 7-robot $R \cup u_i$ can climb and slide around the external surface of S. We use this to move to each u_i, extract it, move to the cell for u'_i and then place a node of the same colour as u_i in it, substituting u_i for a node in R as necessary to create a new 6-robot. By Lemma 5, so long as we approach T_i from R_p, we can climb onto and off T_i to place the nodes using the same movements as the previous theorems. By Lemma 10, placing a black node in the repository cell does not inhibit movement. □

Lemma 14. *Let $W \cup T \cup R$ be the union of the $Stairs$ of an extended staircase W, $T \subseteq \{BRep \cup RRep\}$ from the extended staircase and a 6-node robot R on the cell perimeter of $S \cup T$. Given a shape elimination sequence $\sigma = (u_1, u_2, \ldots, u_n)$ of T, a diagonal line-with-leaves generation sequence σ' which is colour-order preserving w.r.t. σ and a 6-robot placed on the external surface of S, for all $1 \leq i \leq n$ the 6-robot is capable of picking up the node u_i, moving as a 7-robot to u'_i and placing it, and then returning as a 6-robot to u_{i+1}.*

Proof. We follow the procedure of Algorithm 3. By Theorem 1 and Theorem 2, the 6-robot R and 7-robot $R \cup u_i$ can climb and slide around the external surface of $S \cup T$. We use this to move to each u_i, extract it, move to the cell for u'_i and then place a node of the same colour as u_i in it, substituting u_i for a node in R as necessary to create new 6-robot. Since the placement of u'_i is extending $Stairs$, the resulting shape is always orthogonal convex for all $1 \leq i \leq n$. □

Lemma 15. *Let S be a connected orthogonal convex shape. Then there is a connected shape M of 3 nodes (the 3 musketeers) and an attachment of M to the bottom-most row of S, such that $S \cup M$ can reach the configuration D, where D is a diagonal line-with-leaves which is colour-consistent with S.*

Proof. We follow the procedure of Algorithm 1. By Lemma 11 we can form a 6-robot from $S \cup M$. By Lemma 13, we can build an extended staircase from the resulting shape. By Lemma 14, we can then build a diagonal line-with-leaves. Finally, by reversibility, we can place R such that the removal of 3 nodes leaves a larger diagonal line-with-leaves D which is colour-consistent with S. □

Lemma 16. *There exists a connected orthogonal convex shape of n nodes S and a diagonal line-with-leaves T and such that any strategy which transforms S into T requires $O(n^2)$ time steps in the worst case.*

Proof. To construct T, we must transfer nodes using the robot to the anchor node. In the worst case, S is a staircase, and the robot must move nodes from one end to the other. It must therefore make $O(cn^2)$ moves, where c is the maximum number of rotations needed for the robot to move one step. When the extended staircase has been constructed, it must be converted into a diagonal line-with-leaves. In the worst case every column in the staircase has 4 nodes, and the robot must extend *Stairs* until one repository has a single node. Therefore, the robot must make $O(2cn^2)$ moves to travel on both sides of *Stairs*. Combining the worst cases of both procedures therefore takes $O(3cn^2) = O(n^2)$ time steps. □

Theorem 3. *Let S and S' be connected colour-consistent orthogonal convex shapes. Then there is a connected shape M of 3 nodes (the 3 musketeers) and an attachment of M to the bottom-most row of S, such that $S \cup M$ can reach the configuration S' in $O(n^2)$ time steps.*

Proof. By Lemma 15, we can convert S into a diagonal line-with-leaves T. By reversibility, we can convert T into S'. By Lemma 16, this procedure takes $O(n^2)$ time steps. □

6 Conclusions

There are some open problems which follow from the findings of our work. The most obvious is expanding the class of shapes which can be constructed to achieve universal transformation. An example of a bad case is the "double spiral", which is a line forming two connected spirals. In this case, preserving connectivity after the removal of a node requires the robot to get to the centre of a spiral, which may not be possible without a special procedure to "dig" into it without breaking connectivity. Finally, successfully switching to a decentralised model of transformations will greatly expand the utility of the results, especially because most programmable matter systems which model real-world applications implement programs in this way. This in turn could lead to real-world applications for the efficient transformation of programmable matter systems.

References

1. Akitaya, H.A., Arkin, E.M., Damian, M., Demaine, E.D., Dujmović, V., Flatland, R., Korman, M., Palop, B., Parada, I., Renssen, A.v., Sacristán, V.: Universal Reconfiguration of Facet-Connected Modular Robots by Pivots: The O(1) Musketeers. Algorithmica **83**(5), 1316–1351 (May 2021). https://doi.org/10.1007/s00453-020-00784-6
2. Almethen, A., Michail, O., Potapov, I.: Pushing lines helps: Efficient Universal Centralised Transformations for Programmable Matter. Theoretical Computer Science **830-831**, 43–59 (Aug 2020). https://doi.org/10.1016/j.tcs.2020.04.026
3. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M.J., Peralta, R.: Computation in Networks of Passively Mobile Finite-State Sensors. Distributed Computing **18**(4), 235–253 (Mar 2006). https://doi.org/10.1007/s00446-005-0138-3
4. Chen, X., Li, L., Sun, X., Liu, Y., Luo, B., Wang, C., Bao, Y., Xu, H., Peng, H.: Magnetochromatic Polydiacetylene by Incorporation of Fe3O4 Nanoparticles. Angewandte Chemie International Edition **50**(24), 5486–5489 (2011). https://doi.org/10.1002/anie.201100064
5. Connor, M., Michail, O., Potapov, I.: Centralised Connectivity-Preserving Transformations for Programmable Matter: A Minimal Seed Approach. Theoretical Computer Science (2022). https://doi.org/https:doi.org10.1016j.tcs.2022.09.016, `https://www.sciencedirect.com/science/article/pii/S0304397522005527`
6. Daymude, J.J., Derakhshandeh, Z., Gmyr, R., Porter, A., Richa, A.W., Scheideler, C., Strothmann, T.: On the Runtime of Universal Coating for Programmable Matter. Natural Computing **17**(1), 81–96 (Mar 2018). https://doi.org/10.1007/s11047-017-9658-6
7. Derakhshandeh, Z., Dolev, S., Gmyr, R., Richa, A.W., Scheideler, C., Strothmann, T.: Amoebot - a new model for programmable matter. In: Proceedings of the 26th ACM symposium on Parallelism in algorithms and architectures. pp. 220–222. SPAA '14, Association for Computing Machinery, Prague, Czech Republic (Jun 2014). https://doi.org/10.1145/2612669.2612712
8. Derakhshandeh, Z., Gmyr, R., Richa, A.W., Scheideler, C., Strothmann, T.: Universal Shape Formation for Programmable Matter. In: Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures. pp. 289–299. SPAA '16, Association for Computing Machinery, Pacific Grove, California, USA (Jul 2016). https://doi.org/10.1145/2935764.2935784
9. Derakhshandeh, Z., Gmyr, R., Richa, A.W., Scheideler, C., Strothmann, T.: An Algorithmic Framework for Shape Formation Problems in Self-Organizing Particle Systems. In: Proceedings of the Second Annual International Conference on Nanoscale Computing and Communication. pp. 1–2. NANOCOM' 15, Association for Computing Machinery, Boston, MA, USA (Sep 2015). https://doi.org/10.1145/2800795.2800829
10. Doty, D.: Theory of Algorithmic Self-Assembly. Communications of the ACM **55**(12), 78–88 (Dec 2012). https://doi.org/10.1145/2380656.2380675
11. Doty, D.: Timing in Chemical Reaction Networks. In: Proceedings of the 2014 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 772–784. Proceedings, Society for Industrial and Applied Mathematics (Dec 2013). https://doi.org/10.1137/1.9781611973402.57
12. Fekete, S.P., Keldenich, P., Kosfeld, R., Rieck, C., Scheffer, C.: Connected Coordinated Motion Planning with Bounded Stretch. Tech. Rep. arXiv:2109.12381, arXiv (Sep 2021), `http://arxiv.org/abs/2109.12381`, arXiv:2109.12381 [cs] type: article

13. Feldmann, M., Padalkin, A., Scheideler, C., Dolev, S.: Coordinating Amoebots via Reconfigurable Circuits. Journal of Computational Biology: A Journal of Computational Molecular Cell Biology **29**(4), 317–343 (Apr 2022). https://doi.org/10.1089/cmb.2021.0363
14. Gilpin, K., Knaian, A., Rus, D.: Robot Pebbles: One Centimeter Modules for Programmable Matter through Self-Disassembly. In: 2010 IEEE International Conference on Robotics and Automation. pp. 2485–2492 (May 2010). https://doi.org/10.1109/ROBOT.2010.5509817, iSSN: 1050-4729
15. Jordan, C.: Cours d'analyse de l'École polytechnique, vol. 1. Gauthier-Villars et fils (1893)
16. Michail, O., Skretas, G., Spirakis, P.G.: On the Transformation Capability of Feasible Mechanisms for Programmable Matter. Tech. Rep. arXiv:1703.04381, arXiv (Mar 2017), `http://arxiv.org/abs/1703.04381`, arXiv:1703.04381 [cs] type: article
17. Michail, O., Skretas, G., Spirakis, P.G.: On The Transformation Capability of Feasible Mechanisms for Programmable Matter. Journal of Computer and System Sciences **102**, 18–39 (Jun 2019). https://doi.org/10.1016/j.jcss.2018.12.001
18. Michail, O., Spirakis, P.G.: Simple and Efficient Local Codes for Distributed Stable Network Construction. Distributed Computing **29**(3), 207–237 (Jun 2016). https://doi.org/10.1007/s00446-015-0257-4
19. Rothemund, P.W.K.: Folding DNA to Create Nanoscale Shapes and Patterns. Nature **440**(7082), 297–302 (Mar 2006). https://doi.org/10.1038/nature04586
20. Rothemund, P.W.K., Winfree, E.: The Program-size Complexity of Self-Assembled Squares (extended abstract). In: Proceedings of the thirty-second annual ACM symposium on Theory of computing. pp. 459–468. STOC '00, Association for Computing Machinery, Portland, Oregon, USA (May 2000). https://doi.org/10.1145/335305.335358
21. Rubenstein, M., Cornejo, A., Nagpal, R.: Programmable Self-assembly in a Thousand-Robot Swarm. Science **345**(6198), 795–799 (Aug 2014). https://doi.org/10.1126/science.1254295
22. Soloveichik, D., Cook, M., Winfree, E., Bruck, J.: Computation with Finite Stochastic Chemical Reaction Networks. Natural Computing **7**(4), 615–633 (Dec 2008). https://doi.org/10.1007/s11047-008-9067-y
23. Thalamy, P., Piranda, B., Bourgeois, J.: Distributed Self-Reconfiguration using a Deterministic Autonomous Scaffolding Structure. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems. pp. 140–148. AAMAS '19, International Foundation for Autonomous Agents and Multiagent Systems, Montreal QC, Canada (May 2019)
24. Thalamy, P., Piranda, B., Bourgeois, J.: 3D Coating Self-Assembly for Modular Robotic Scaffolds. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 11688–11695 (Oct 2020). https://doi.org/10.1109/IROS45743.2020.9341324, iSSN: 2153-0866
25. Winfree, E.: Algorithmic Self-assembly of DNA. Ph.D. thesis, California Institute of Technology (Jun 1998)
26. Woods, D., Chen, H.L., Goodfriend, S., Dabby, N., Winfree, E., Yin, P.: Active Self-Assembly of Algorithmic Shapes and Patterns in Polylogarithmic Time. In: Proceedings of the 4th conference on Innovations in Theoretical Computer Science. pp. 353–354. ITCS '13, Association for Computing Machinery, Berkeley, California, USA (Jan 2013). https://doi.org/10.1145/2422436.2422476