On the Transformation Capability of Feasible Mechanisms for Programmable Matter*

Othon Michail[1], George Skretas[2], and Paul G. Spirakis[3]

1 **Department of Computer Science, University of Liverpool, UK**
`Othon.Michail@liverpool.ac.uk`

2 **Computer Engineering and Informatics Department, Patras University, Greece**
`skretas@ceid.upatras.gr`

3 **Department of Computer Science, University of Liverpool, UK & CTI & Computer Engineering and Informatics Department, Patras University, Greece**
`P.Spirakis@liverpool.ac.uk`

Abstract

In this work, we study theoretical models of *programmable matter* systems. The systems under consideration consist of spherical modules, kept together by magnetic forces and able to perform two minimal mechanical operations (or movements): *rotate* around a neighbor and *slide* over a line. In terms of modeling, there are n nodes arranged in a 2-dimensional grid and forming some initial *shape*. The goal is for the initial shape A to *transform* to some target shape B by a sequence of movements. Most of the paper focuses on *transformability* questions, meaning whether it is in principle feasible to transform a given shape to another. We first consider the case in which only rotation is available to the nodes. Our main result is that deciding whether two given shapes A and B can be transformed to each other is in **P**. We then insist on rotation only and impose the restriction that the nodes must maintain global connectivity throughout the transformation. We prove that the corresponding transformability question is in **PSPACE** and study the problem of determining the minimum *seeds* that can make feasible otherwise infeasible transformations. Next we allow both rotations and slidings and prove universality: any two connected shapes A, B of the same number of nodes, can be transformed to each other without breaking connectivity. The worst-case number of movements of the generic strategy is $\Theta(n^2)$. We improve this to $O(n)$ parallel time, by a pipelining strategy, and prove optimality of both by matching lower bounds. We next turn our attention to distributed transformations. The nodes are now distributed processes able to perform communicate-compute-move rounds. We provide distributed algorithms for a general type of transformation.

1998 ACM Subject Classification F.1.1 Models of Computation; F.1.2 Modes of Computation; F.1.3 Complexity Measures and Classes; F.2.2 Nonnumerical Algorithms and Problems; I.2.9 Robotics

Keywords and phrases programmable matter, transformation, reconfigurable robotics, shape formation, complexity, distributed algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2017.

* Supported in part by the School of EEE/CS of the University of Liverpool, NeST initiative. The full version of this paper (including all omitted details and a number of helpful illustrations) can be found at: https://arxiv.org/abs/1703.04381

44th International Colloquium on Automata, Languages, and Programming (ICALP 2017).
Editors: Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl;
Article No. ; pp. :1–:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Programmable matter refers to any type of matter that can *algorithmically* change its physical properties. For a concrete example, imagine a material formed by a collection of spherical nanomodules kept together by magnetic forces. Each module is capable of storing (in some internal representation) and executing a simple program that handles communication with nearby modules and that controls the module's electromagnets, in a way that allows the module to *rotate* or *slide* over neighboring modules. Such a material would be able to adjust its *shape* in a programmable way. Other examples of physical properties of interest for real applications would be connectivity, color [25, 5], and strength of the material.

There are already some first impressive outcomes towards the development of programmable materials (even though it is evident that there is much more work to be done in the direction of real systems), such as programmed DNA molecules that self-assemble into desired structures [28, 14] and large collectives of tiny identical robots that orchestrate resembling a single multi-robot organism (e.g., the Kilobot system [30]). Other systems for programmable matter include [21, 23]. Ambitious long-term applications of programmable materials include molecular computers, collectives of nanorobots injected into the human circulatory system for monitoring and treating diseases, or even self-reproducing and self-healing machines (see also [27]).

Apart from the fact that systems work is still in its infancy, there is also an apparent lack of unifying formalism and theoretical treatment. Still there are some first theoretical efforts aiming at understanding the fundamental possibilities and limitations of this prospective. The area of *algorithmic self-assembly* tries to understand how to program molecules (mainly DNA strands) to manipulate themselves, grow into machines and at the same time control their own growth [14]. The theoretical model guiding the study in algorithmic self-assembly is the Abstract Tile Assembly Model (aTAM) [33, 29] and variations. Recently, a model, called the *nubot* model, was proposed for studying the complexity of self-assembled structures with active molecular components [34]. Another very recent model, called the *Network Constructors* model, studied what stable networks can be constructed by a population of finite-automata that interact randomly like molecules in a well-mixed solution and can establish bonds with each other according to the rules of a common small protocol [26]. The development of Network Constructors was based on the *Population Protocol* model of Angluin *et al.* [2], that does not include the capability of creating bonds and focuses more on the computation of functions on inputs. A very interesting fact about population protocols is that they are formally equivalent to *chemical reaction networks* (CRNs), "which model chemistry in a *well-mixed solution* and are widely used to describe information processing occurring in natural cellular regulatory networks" [15]. Also the recently proposed *Amoebot* model, "offers a versatile framework to model self-organizing particles and facilitates rigorous algorithmic research in the area of programmable matter" [10, 12, 11, 13]. Other related work includes mobile and reconfigurable robotics [6, 24, 31, 20, 32, 8, 7, 4, 36, 1, 35], puzzles [9, 22], and passive systems [2, 3, 26, 19, 33, 29].

It seems that the right way for theory to boost the development of more refined real systems is to reveal the *transformation capabilities of mechanisms and technologies that are available now*, rather than by exploring the unlimited variety of theoretical models that are not expected to correspond to a real implementation in the near future. In this paper, we follow such an approach, by studying the transformation capabilities of models for programmable matter, which are based on minimal mechanical capabilities, easily implementable by existing technology.

1.1 Our Approach

We study a minimal programmable matter system consisting of n cycle-shaped modules, with each module (or *node*) occupying at any given time a cell of the 2-dimensional (abbreviated "2D" throughout) grid (no two nodes can occupy the same cell at the same time). Therefore, the composition of the programmable matter systems under consideration is discrete. Our main question throughout is whether an initial arrangement of the material can *transform* (either in principle, e.g., by an external authority, or by itself) to some other target arrangement. In more technical terms, we are provided with an *initial shape* A and a *target shape* B and we are asked whether A *can be transformed to* B via a sequence of *valid* transformation steps. Usually, a step consists either of a *valid movement* of a single node (in the *sequential case*) or of more than one nodes at the same time (in the *parallel case*). We consider two quite primitive types of movement. The first one, called *rotation*, allows a node to rotate 90° around one of its neighbors either clockwise or counterclockwise and the second one, called *sliding*, allows a node to slide by one position "over" two neighboring nodes. Both movements succeed only if the whole direction of movement is free of obstacles (i.e., other nodes blocking the way). More formal definitions are provided in Section 2. One part of the paper focuses on the case in which only rotation is available to the nodes and the other part studies the case in which both rotation and sliding are available. The latter case has been studied to some extent in the past in the, so called, *metamorphic systems* [17, 18, 16], which makes those studies the closest to our approach.

For rotation only, we introduce the notion of *color-consistency* and prove that if two shapes are not color-consistent then they cannot be transformed to each other. On the other hand, color-consistency does not guarantee transformability, as there is an infinite set of pairs (A, B) such that A and B are color consistent but still they cannot be transformed to each other. At this point, observe that if A can be transformed to B then the inverse is also true, as all movements considered in this paper are *reversible*. We distinguish two main types of transformations: those that are allowed to break the connectivity of the shape during the transformation and those that are not; we call the corresponding problems Rot-Transformability and RotC-Transformability, respectively. Our main result regarding Rot-Transformability is that Rot-Transformability $\in$ **P**. To prove polynomial-time decidability, we prove that two connected shapes A and B of the same order (i.e., having the same number of nodes) are transformable to each other iff both have at least one movement available. Therefore, transformability reduces to checking the availability of a movement in the initial and target shapes.

We next study RotC-Transformability, in which again the only available movement is rotation, but now connectivity of the material has to be preserved throughout the transformation. The property of preserving the connectivity is expected to be a crucial property for programmable matter systems, as it allows the material to maintain coherence and strength, to eliminate the need for wireless communication, and, finally, enables the development of more effective power supply schemes, in which the modules can share resources or in which the modules have no batteries but are instead constantly supplied with energy by a centralized source (or by a supernode that is part of the material itself). Such benefits can lead to simplified designs and potentially to reduced size of individual modules. We first prove that RotC-Transformability $\in$ **PSPACE**. The rest of our results here are strongly based on the notion of a *seed*. This stems from the observation that a large set of infeasible transformations become feasible by introducing to the initial shape an additional, and usually quite small, seed; i.e., a small shape that is being attached to some point of the initial shape. We investigate seeds that could serve as components capable of traveling the perimeter of an arbitrary connected shape A. Such seed-shapes are very convenient as they are capable of "simulating" the *universal transformation* techniques that are possible if we have both rotation and sliding movements available (discussed in the sequel). To this end, we prove that all seeds of size ≤ 4 cannot serve for this purpose, by proving that they cannot even walk the perimeter of a simple line shape. On the other hand, we manage to show that a *6-seed* succeeds, and this provides a first indication, that there might be a large family of shapes that can be transformed to each other with rotation only and without breaking connectivity.

Next, we consider the case in which both rotation and sliding are available and insist on connectivity preservation. We first provide a proof that this combination of simple movements is *universal* w.r.t. transformations, as any pair of connected shapes A and B of the same order can be transformed to each other without ever breaking the connectivity throughout the transformation (a first proof of this fact had already appeared in [16]). This generic transformation requires $\Theta(n^2)$ sequential movements in the worst case. By a potential-function argument we show that no transformation can improve on this worst-case complexity for some specific pairs of shapes and this lower bound is independent of connectivity preservation; it only depends on the inherent *transformation-distance* between the shapes. To improve on this, either some sort of parallelism must be employed or more powerful movement mechanisms, e.g., movements of whole sub-shapes in one step. We investigate the former approach, and prove that there is a *pipelining* general transformation strategy that improves the time to $O(n)$ (parallel time). We also give a matching $\Omega(n)$ lower bound. On the way, we also show that this parallel complexity is feasible even if the nodes are labeled, meaning that individual nodes must end up in specific positions of the target-shape.

Finally, we assume that the nodes are distributed processes able to perform communicate-compute-move rounds (where, again, both rotation and sliding movements are available) and provide distributed algorithms for a general type of transformation.

Section 2 brings together all definitions and basic facts that are used throughout the paper. In Section 3, we study programmable matter systems equipped only with rotation movement. In Section 4, we insist on rotation only, but additionally require that the material maintains connectivity throughout the transformation. In Section 5, we investigate the combined effect of rotation and sliding movements. Finally, in Section 6 we conclude and give further research directions that are opened by our work.

2 Preliminaries

The programmable matter systems considered in this paper operate on a 2D square grid, with each position (or *cell*) being uniquely referred to by its $y \geq 0$ and $x \geq 0$ coordinates. Such a system consists of a set V of n *modules*, called *nodes* throughout. Each node may be viewed as a spherical module fitting inside a cell of the grid. At any given time, each node $u \in V$ occupies a cell $o(u) = (o_y(u), o_x(u)) = (i, j)$ (where i corresponds to a row and j to a column of the grid) and no two nodes may occupy the same cell. At any given time t, the positioning of nodes on the grid defines an undirected *neighboring relation* $E(t) \subset V \times V$, where $\{u, v\} \in E$ iff $o_y(u) = o_y(v)$ and $|o_x(u) - o_x(v)| = 1$ or $o_x(u) = o_x(v)$ and $|o_y(u) - o_y(v)| = 1$, that is, if u and v are either *horizontal* or *vertical* neighbors on the grid, respectively. A more informative way to define the system at a given time t, and thus often more convenient, is as a mapping $P_t \colon \mathbb{N}_{\geq 0} \times \mathbb{N}_{\geq 0} \to \{0, 1\}$ where $P_t(i, j) = 1$ iff cell (i, j) is occupied by a node.

At any given time t, $P_t^{-1}(1)$ defines a *shape*. Such a shape is called *connected* if $E(t)$

defines a connected graph. A connected shape is called *convex* if for any two occupied cells, the line that connects their centers does not pass through an empty cell. We call a shape *discrete-convex* if for any two occupied cells, belonging either to the same row or the same column, the line that connects their centers does not pass through an empty cell; i.e., in the latter we exclude diagonal lines. We call a shape *compact* if it has no holes.

In general, shapes can *transform* to other shapes via a sequence of one or more *movements* of individual nodes. Time consists of discrete *steps* (or *rounds*) and in every step, zero or more movements may occur. In the *sequential* case, at most one movement may occur per step, and in the *parallel* case any number of "valid" movements may occur in parallel. [1] We consider two types of movements: (i) *rotation* and (ii) *sliding*. In both movements, a single node moves relative to one or more neighboring nodes as we just explain.

A single *rotation* movement of a node u is a 90° *rotation* of u around one of its neighbors. Let (i, j) be the current position of u and let its neighbor be v occupying the cell $(i-1, j)$ (i.e., lying below u). Then u can *rotate* 90° clockwise (counterclockwise) around v iff the cells $(i, j+1)$ and $(i-1, j+1)$ ($(i, j-1)$ and $(i-1, j-1)$, respectively) are both empty. By rotating the whole system 90°, 180°, and 270°, all possible rotation movements are defined analogously. A single *sliding* movement of a node u is a one-step horizontal or vertical movement "over" a horizontal or vertical line of (neighboring) nodes of length 2. In particular, if (i, j) is the current position of u, then u can *slide* rightwards to position $(i, j+1)$ iff $(i, j+1)$ is not occupied and there exist nodes at positions $(i-1, j)$ and $(i-1, j+1)$ or at positions $(i+1, j)$ and $(i+1, j+1)$, or both. Precisely the same definition holds for up, left, and down sliding movements by rotating the whole system 90°, 180°, and 270° counterclockwise, respectively.

Let A and B be two shapes. We say that A *transforms to* B *via a movement* m (which can be either a rotation or a sliding), denoted $A \xrightarrow{m} B$, if there is a node u in A such that if u applies m, then the shape resulting after the movement is B (possibly after rotations and translations of the resulting shape, depending on the application). 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 $A \xrightarrow{m} B$ for some movement m. We say that A *transforms* to B (or that B is *reachable* from A) and write $A \rightsquigarrow B$, if there is a sequence of shapes $A = C_0, C_1, \dots, C_t = B$, such that $C_i \to C_{i+1}$ for all i, $0 \le i < t$. We should mention that we do not always allow m to be any of the two possible movements. In particular, in Sections 3 and 4 we only allow m to be a rotation, as we there restrict attention to systems in which only rotation is available. We shall clearly explain what movements are permitted in each part of the paper. Observe now that both rotation and sliding are *reversible* movements, a fact that we extensively use in our results. Based on this, it can be proved that the relation '$\rightsquigarrow$' is a partial equivalence relation. When the only available movement is rotation, there are shapes in which no rotation can be performed. If we introduce a *null* rotation, then every shape may transform to itself by applying the *null* rotation, and '$\rightsquigarrow$' becomes an equivalence relation.

The following are the main transformation problems that are considered in this work:

Rot-Transformability. Given an initial shape A and a target shape B (usually both connected), decide whether A can be transformed to B (usually, under translations and rotations of the shapes) using only a sequence of rotation movements.

RotC-Transformability. Special case of Rot-Transformability, where A and B are connected shapes and connectivity must be preserved throughout the transformation.

RS-Transformability. Variant of Rot-Transformability in which both rotation and

[1] By "valid", we mean here subject to the constraint that their whole movement paths correspond to pairwise disjoint sub-areas of the grid.

sliding movements are available.
Minimum-Seed-Determination. Given an initial shape A and a target shape B determine a minimum-size seed and an initial positioning of that seed relative to A that makes the transformation from A to B feasible.

3 Rotation

In this section, the only permitted movement is 90° *rotation* around a neighbor. Our main result in this section is that Rot-Transformability $\in$ **P**.

Consider a black and red checkered coloring of the 2D grid. Any shape S may be viewed as a colored shape consisting of $b(S)$ blacks and $r(S)$ reds. Call two shapes A and B *color-consistent* if $b(A) = b(B)$ and $r(A) = r(B)$ and call them *color-inconsistent* otherwise. Call a transformation from a shape A to a shape C *color-preserving* if A and C are color consistent.

▶ Observation 1. The rotation movement is color-preserving. Formally, $A \rightsquigarrow C$ (restricted to rotation only) implies that A and C are color-consistent. In particular, every node beginning from a black (red) position of the grid, will always be on black (red, respectively) positions throughout a transformation.

Based on this property of the rotation movement, we may call each node *black* or *red* throughout a transformation, based only on its initial coloring. Observation 1 gives a partial way to determine that two shapes A and B cannot be transformed to each other by rotations.

▶ Proposition 1. If two shapes A and B are color-inconsistent, then it is impossible to transform one to the other by rotations only.

▶ Proposition 2. There is a generic connected shape, called *line-with-leaves*, that has a color-consistent version for any connected shape A.

Proof. Let red be the majority color of A and k be the number of black nodes of A. Consider a bi-color line starting with a black node and ending to a black node, such that all k blacks are exhausted. To do this, $k - 1$ reds are needed in order to alternate blacks and reds on the line. Since A is connected, it can have at most $3k + 1$ reds. By adding red leaf-nodes around the blacks of the line, we can achieve the whole range of possible number of reds, from k to $3k + 1$. ◀

Based on this, we now show that the inverse of Proposition 1 is not true, that is, it does not hold that any two color-consistent shapes can be transformed to each other by rotations.

▶ Proposition 3. There is an infinite set of pairs (A, B) of connected shapes, such that A and B are color-consistent but cannot be transformed to each other by rotations only.

Proof. For shape A, take a rhombus in which no node is able to rotate. By Proposition 2, any such A has a color-consistent shape B from the family of line-with-leaves shapes, such that $B \neq A$. We conclude that A and B are distinct color-consistent shapes which cannot be transformed to each other, and there is an infinite number of such pairs, as the number of black nodes of A can be made arbitrarily large. ◀

Propositions 1 and 3 give a partial characterization of pairs of shapes that cannot be transformed to each other. Observe that the impossibilities proved so far, hold for all possible transformations based on rotation only, including those that are allowed to break connectivity.

The next theorem states that the inclusion between RotC-Transformability and Rot-Transformability is strict, that is, there are strictly more feasible transformations if we allow connectivity to break. We prove this by showing that there is a feasible transformation, namely folding a spanning line in half, in Rot-Transformability\RotC-Transformability.

▶ **Theorem 1.** RotC-Transformability $\subset$ Rot-Transformability.

Aiming at a general transformation, we ask whether there is some minimal addition to a shape that would allow it to transform. The solution turns out to be as small as a *2-line seed* (a bi-color pair, usually referred to as "2-line" or "2-seed") lying initially somewhere "outside" the boundaries of the shape. Based on the above assumptions, we prove that any pair of color-consistent connected shapes A and B can be transformed to each other. The idea is to exploit the fact that the 2-line can move freely in any direction and to use it in order to extract from A another 2-line. In this way, a 4-line seed is formed, which can also move freely in all directions. Then we use the 4-line as a transportation medium for carrying the nodes of A, one at a time. We exploit these mobility mechanisms to transform A into a uniquely defined shape from the line-with-leaves family of Proposition 2. But if any connected shape A with an extra 2-line can be transformed to its color-consistent line-with-leaves version with an extra 2-line, then this also holds inversely due to reversibility, and it follows that any A can be transformed to any B by transforming A to its line-with-leaves version L_A and then inverting the transformation from B to $L_B = L_A$.

▶ **Theorem 2.** *If connectivity can break and there is a 2-line seed provided "outside" the initial shape, then any pair of color-consistent connected shapes A and B can be transformed to each other by rotations only.*

Proof. Without loss of generality (due to symmetry and the 2-line's unrestricted mobility), it suffices to assume that the seed is provided somewhere below the lowest row l occupied by the shape A. We show how A can be transformed to L_A with the help of the seed. We define L_A as follows: Let k be the cardinality of the minority color, let it be the black color. As there are at least k reds, we can create a horizontal line of length $2k$, i.e., $u_1, u_2, \ldots, u_{2k}$, starting with a black (i.e., u_1 is black), and alternating blacks and reds. In this way, the blacks are exhausted. The remaining $\leq (3k+1) - k = 2k+1$ reds are then added as leaves of the black nodes, starting from the position to the left of u_1 and continuing counterclockwise, i.e., below u_1, below u_3, ..., below u_{2k-1}, above u_{2k-1}, above u_{2k-3}, and so on. This gives the same shape from the line-with-leaves family, for all color-consistent shapes (observe that the leaf to the right of the line is always placed). L_A shall be constructed on rows $l-5$ to $l-3$ (not necessarily inclusive), with u_1 on row $l-4$ and a column j preferably between those that contain A.

First, extract a 2-line from A, from row l, so that the 2-line seed becomes a 4-line seed. To see that this is possible for every shape A of order at least 2, distinguish the following two cases: (i) If the lowest row has a horizontal 2-line, then the 2-line can leave the shape without any help and approach the 2-seed. (ii) If not, then take any node u of row l. As A is connected and has at least two nodes, u must have a neighbor v above it. The only possibility that the 2-line u,v is not free to leave A is when v has both a left and a right neighbor, but this can be resolved with the help of the 2-line.

To transform A to L_A, given the 4-line seed, do the following:

- While blacks is still present in A:
  - If on the current lowest row occupied by A, there is a 2-line that can be extracted alone and moved towards L_A, then perform the shortest such movement that attaches the 2-line to the right endpoint of L_A's line $u_1, u_2, \ldots$.
  - If not, then do the following. Maintain a *repository* of nodes at the empty space below row $l-7$, initially empty. If, either in the lowest row of A or in the repository, there

is a node of opposite color than the current color of the right endpoint of L_A's line, use the 4-line to transfer such a node and make it the new right endpoint of L_A's line. Otherwise, use the 4-line to transfer a node of the lowest row of A to the repository.

- Once black has been exhausted from A and the repository (i.e., when u_{2k-3} has been placed; u_{2k-1} and u_{2k} will only be placed in the end as they are part of the 4-line), transfer a red to position u_{2k-2}. If there are no more nodes left, run the termination phase, otherwise transfer the remaining nodes (all red) with the 4-line, one after the other, and attach them as leaves around the blacks of L_A's line, beginning from the position to the left of u_1 counterclockwise, as described above (skipping position u_{2k}).
- Termination phase: the line-with-leaves is ready, apart from positions u_{2k-1}, u_{2k} which require a 2-line from the 4-line. If the position above u_{2k-1} is empty, then extract a 2-line from the 4-line and transfer it to the positions u_{2k-1}, u_{2k}. This completes the transformation. If the position above u_{2k-1} is occupied by a node u_{2k+1}, then place the whole 4-line vertically with its lowest endpoint on u_{2k}. Then rotate the top endpoint counterclockwise, to move above u_{2k+1}, then rotate u_{2k+1} clockwise around it to move to its left, then rotate the node above u_{2k} counterclockwise to move to u_{2k-1}, and finally restore u_{2k+1} to its original position. This completes the construction (the 2-line that always remains can be transferred in the end to a predefined position).

◀

The natural next question is to what extent the 2-line seed assumption can be dropped. Clearly, by Proposition 3, this cannot be always possible. The following lemma gives a sufficient and necessary condition for dropping the 2-line seed assumption.

▶ **Lemma 3.** *A 2-seed can be extracted from a shape iff a single rotation move is available on the shape.*

▶ **Theorem 4.** ROT-TRANSFORMABILITY $\in$ **P**.

Proof. If the two connected input shapes of the same order are not already equal, then, by Lemma 3 and Theorem 2, it suffices to check if both shapes have an available movement. If yes, *accept*, otherwise, *reject*. These checks can be easily performed in polynomial time. ◀

4 Rotation and Connectivity Preservation

In this section, we restrict our attention to transformations that transform a connected shape A to one of its color-consistent connected shapes B, without ever breaking the connectivity of the shape on the way. As already mentioned in the introduction, connectivity preservation is a very desirable property for programmable matter, as, among other positive implications, it guarantees that communication between all nodes is maintained, it minimizes transformation failures, requires less sophisticated actuation mechanisms, and increases the external forces required to break the system apart.

We begin by proving that ROTC-TRANSFORMABILITY can be decided in deterministic polynomial space.

▶ **Theorem 5.** ROTC-TRANSFORMABILITY $\in$ **PSPACE**.

As already shown in Theorem 1, the connectivity-preservation constraint increases the class of infeasible transformations. A convenient turnaround in such cases, is to introduce a suitable seed that can assist the transformation. For example, we can circumvent the impossibility of folding a line $u_1, u_2, \ldots, u_n$ in half, by adding a 3-line seed v_1, v_2, v_3, horizontally aligned over nodes u_3, u_4, u_5 of the line. Interestingly, adding the seed over nodes u_4, u_5, u_6 does not work. Therefore, the problem that we face in such cases, is to find a minimum seed (could be any connected small shape, not necessarily a line) and a placement of that seed, that enables the otherwise infeasible transformation (*Minimum-Seed-Determination* problem). In the rest of this section, we try to identify a minimum seed that can walk the perimeter of any shape, hoping that it will be able to move nodes gradually to a predetermined position, in order to transform the initial shape into a line-with-leaves (as in Theorem 2, but without ever breaking connectivity this time). [2]

▶ **Theorem 6.** *If connectivity must be preserved: (i) Any* (≤ 4)*-seed cannot traverse the perimeter of a line, (ii) A 6-seed can traverse the perimeter of any discrete-convex shape.*

5 Rotation and Sliding

In this section, we study the combined effect of rotation and sliding movements. We begin by proving that rotation and sliding together are *transformation-universal*, meaning that they can transform any given shape to any other shape of the same size without ever breaking the connectivity during the transformation.

▶ **Theorem 7.** *Let A and B be any connected shapes, such that $|A| = |B| = n$. Then A and B can be transformed to each other by rotations and slidings, without breaking the connectivity during the transformation.*

Proof. It suffices to show that any connected shape A can be transformed to a spanning line L using only rotations and slidings and without breaking connectivity during the transformation. If we show this, then A can be transformed to L and B can be transformed to L (as A and B have the same order, therefore corresponding to the same spanning line L), and by reversibility of these movements, A and B can be transformed to each other via L.

Pick the rightmost column of the grid containing at least one node of A, and consider the lowest node of A in that column. Call that node u. Observe that all cells to the right of u are empty. Let the cell of u be (i, j). The final constructed line will start at (i, j) and end at $(i, j + n - 1)$.

The transformation is partitioned into $n-1$ phases. In each phase k, we pick a node from the original shape and move it to position $(i, j + k)$, that is, to the right of the right endpoint of the line formed so far. In phase 1, position $(i, j + 1)$ is a cell of the perimeter of A. So, even if it happens that u is a node of degree 1, it can be proved that there must be another such node $v \in A$ that can walk the whole perimeter of $A' = A - \{v\}$. As $u \neq v$, $(i, j + 1)$ is also part of the perimeter of A', therefore, v can move to $(i, j + 1)$ by rotations and slidings. But A' is connected, $A' \cup \{(i, j + 1)\}$ is also connected, and also all intermediate shapes were connected, because v moved on the perimeter and, therefore, it never disconnected from the rest of the shape during its movement.

In general, the transformation preserves the following invariant. At the beginning of phase k, $1 \leq k \leq n - 1$, there is a connected shape $S(k)$ (where $S(1) = A$) to the left of of column j (j inclusive) and a line of length $k - 1$ starting from position $(i, j + 1)$ and growing to the right. Restricting attention to $S(k)$, there is always a $v \neq u$ that could (hypothetically) move to position $(i, j + 1)$ if it were not occupied. This implies that before the final movement that

[2] Another way to view this, is as an attempt to simulate the universal transformations based on combined rotation and sliding (presented in Section 5), in which single nodes are able to walk the perimeter of the shape.

would place v on $(i, j+1)$, v must have been in $(i+1, j)$ or $(i+1, j+1)$, if we assume that v always walks in the clockwise direction. Observe now that from each of these positions v can perform zero or more right slidings above the line in order to reach the position above the right endpoint $(i, j+k-1)$ of the line. When this occurs, a final clockwise rotation makes v the new right endpoint of the line. The only exception is when v is on $(i+1, j+1)$ and there is no line to the right of (i, j) (this implies the existence of a node on $(i+1, j)$, otherwise connectivity of $S(k)$ would have been violated). In this case, v just performs a single downward sliding to become the right endpoint of the line. ◀

▶ **Theorem 8.** *The transformation of Theorem 7 requires $\Theta(n^2)$ movements in the worst case.*

Theorem 8 shows that the above generic strategy is slow in some cases, as is the case of transforming a staircase shape into a spanning line. A *staircase* is defined as a shape of the form $(i, j), (i-1, j), (i-1, j+1), (i-2, j+1), (i-2, j+2), (i-3, j+2), \ldots$. We shall now show that there are pairs of shapes for which any strategy and not only this particular one, may require a quadratic number of steps to transform one shape to the other.

▶ **Definition 9.** Define the *potential of a shape* A as its minimum "distance" from the line L, where $|A| = |L|$. The *distance* is defined as follows: Consider any placement of L relative to A and any pairing of the nodes of A to the nodes of the line. Then sum up the Manhattan distances [3] between the nodes of each pair. The minimum sum between all possible relative placements and all possible pairings is the distance between A and L and also A's potential.

Observe that the potential of the line is 0 as it can be totally aligned on itself and the sum of the distances is 0.

▶ **Lemma 10.** *The potential of a staircase is $\Theta(n^2)$.*

Proof. We prove it for horizontal placement of the line, as the vertical case is symmetric. Any such placement leaves either above or below it at least half of the nodes of the staircase (maybe minus 1). W.l.o.g. let it be above it. Every two nodes, the height increases by 1, therefore there are 2 nodes at distance 1, 2 at distance 2,. . ., 2 at distance n/4. Any matching between these nodes and the nodes of the line gives for every pair a distance at least as large as the vertical distance between the staircase's node and the line, thus, the total distance is at least $2 \cdot 1 + 2 \cdot 2 + \ldots + 2 \cdot (n/4) = 2 \cdot (1 + 2 + \ldots + n/4) = (n/4) \cdot (n/4 + 1) = \Theta(n^2)$. We conclude that the potential of the staircase is $\Theta(n^2)$. ◀

▶ **Theorem 11.** *Any transformation strategy based on rotations and slidings which performs a single movement per step requires $\Theta(n^2)$ steps to transform a staircase into a line.*

Proof. To show that $\Omega(n^2)$ movements are needed to transform the staircase into a line, it suffices to observe that the difference in their potentials is that much and that one rotation or one sliding can decrease the potential by at most 1. ◀

▶ Remark. The above lower bound is independent of connectivity preservation. It is just a matter of the total distance based on single distance-one movements.

Finally, it is interesting to observe that such lower bounds can be computed in polynomial time, because there is a polynomial-time algorithm for computing the distance between two shapes.

[3] The Manhattan distance between two points (i, j) and (i', j') is given by $|i - i'| + |j - j'|$.

▶ Proposition 4. Let A and B be connected shapes. Then their distance $d(A, B)$ can be computed in polynomial time.

To give a faster transformation either pipelining must be used (allowing for more than one movement in parallel) or more complex mechanisms that move sub-shapes consisting of many nodes, in a single step. We follow the former approach, by allowing an unbounded number of rotation and/or sliding movements to occur simultaneously in a single step (though, in pairwise disjoint areas).

▶ Proposition 5. There is a pipelining strategy that transforms a staircase into a line in $O(n)$ parallel time.

Proof. Number the nodes of the staircase 1 through n starting from the top and following the staircase's connectivity until the bottom-right node is reached. These gives an odd-numbered upper diagonal and an even-numbered lower diagonal. Node 1 moves as in Theorem 7. Any even node w starts moving as long as its upper odd neighbor has reached the same level as w (e.g., node 2 first moves after node 1 has arrived to the right of node 3). Any odd node $z > 1$ starts moving as long as its even left neighbor has moved one level down (e.g., node 3 first moves after node 2 has arrived to the right of 5). After a node starts moving, it moves in every step as in Theorem 7 (but now many nodes can move in parallel, implementing a pipelining strategy). It can be immediately observed that any node i starts after at most 3 movements of node $i-1$ (actually, only 2 movements for even i), so after, roughly, at most $3n$ steps, node $n-2$ starts. Moreover, a node that starts, arrives at the right endpoint of the line after at most n steps, which means that after at most $4n = O(n)$ steps, all nodes have taken their final position in the line. ◀

Proposition 5 gives a hint that pipelining could be a general strategy to speed-up transformations. We next show how to generalize this technique to any possible pair of shapes.

▶ **Theorem 12.** *Let A and B be any connected shapes, such that $|A| = |B| = n$. Then there is a pipelining strategy that can transform A to B (and inversely) by rotations and slidings, without breaking the connectivity during the transformation, in $O(n)$ parallel time.*

Proof. The transformation is a pipelined version of the sequential transformation of Theorem 7. Now, instead of picking an arbitrary next candidate node of $S(k)$ to walk the perimeter of $S(k)$ clockwise, we always pick the rightmost clockwise node $v_k \in S(k)$, that is, the node that has to walk the shortest clockwise distance to arrive at the line being formed. This implies that the subsequent candidate node v_{k+1} to walk is always "behind" v_k in the clockwise direction and is either already free to move or is enabled after v_k's departure. Observe that after at most 3 clockwise movements, v_k can no longer be blocking v_{k+1} on the (possibly updated) perimeter. Moreover, the clockwise move of v_{k+1} only introduces a gap in its original position, therefore it only affects the structure of the perimeter "behind" it. The strategy is to start the walk of node v_{k+1} as soon as v_k is no longer blocking its way. As in Proposition 5, once a node starts, it moves in every step, and again any node arrives at the end of the forming line after at most n movements. It follows that if the pipelined movement of nodes cannot be blocked in any way, after $4n = O(n)$ steps all nodes must have arrived at their final positions. Observe now that the only case in which pipelining could be blocked is when a node is sliding through a (necessarily dead-end) "tunnel" of height 1. To avoid this, the nodes shortcut the tunnel, by visiting only its first position (i, j) and then simply skipping the whole walk inside it (that walk would just return them to position (i, j) after a number of steps). ◀

We next show that even if A and B are labeled shapes, that is, their nodes are assigned the indices $1, \dots, n$ (uniquely, i.e., without repetitions), we can still transform the labeled A to the labeled B with only a linear increase in parallel time. We only consider transformations in which the nodes never change indices in any way (e.g., cannot transfer them, or swap them), so that each particular node of A must eventually occupy (physically) a particular position of B (the one corresponding to its index).

▶ **Corollary 13.** *The labeled version of the transformation of Theorem 12 can be performed in $O(n)$ parallel time.*

An immediate observation is that a linear-time transformation does not seem satisfactory for all pairs of shapes. To this end, take a square S and rotate its top-left corner u, one position clockwise, to obtain an almost-square S'. Even though, a single counter-clockwise rotation of u suffices to transform S' to S, the transformation of Theorem 12 may go all the way around and first transform S' to a line and then transform the line to S. In this particular example, the distance between S and S', according to Definition 9, is 2, while the generic transformation requires $\Theta(n)$ parallel time. So, it is plausible to ask if any transformation between two shapes A and B can be performed in time that grows as a function of their distance $d(A, B)$. We show that this cannot always be the case, by presenting two shapes A and B with $d(A, B) = 2$, such that A and B require $\Omega(n)$ parallel time to be transformed to each other.

▶ Proposition 6. There are two shapes A and B with $d(A, B) = 2$, such that A and B require $\Omega(n)$ parallel time to be transformed to each other.

In the full version, we also study the RS-Transformability problem in distributed systems and give an algorithm that transforms a large family of shapes into a spanning line:

▶ **Theorem 14.** *We provide an algorithm, called* Compact Line*, that can transform any compact shape into a spanning line.*

6 Conclusions and Further Research

There are many open problems related to the findings of the present work. First, a compromise could be to allow some restricted degree of connectivity breaking. There are other meaningful "good" properties that we would like to maintain throughout a transformation, like the *strength* of the shape.

Transformation seems in general harder if we restrict the maximum area or dimensions during its course. Also, restricting the boundaries gives models equivalent to several interesting puzzles, like the famous 15-puzzle. Techniques developed in the context of puzzles could prove valuable for analyzing and characterizing discrete programmable matter systems.

We intentionally restricted attention to very minimal actuation mechanisms. More sophisticated mechanical operations would enable a larger set of transformations and possibly also reduce the time complexity. Such an example is the ability of a node to become inserted between two neighboring nodes.

There are also some promising specific technical questions: What is the exact complexity of RotC-Transformability? What is the complexity of computing the optimum transformation? Can it be satisfactorily approximated? Finally, regarding the distributed transformations, there are various interesting variations of the model considered here, that would make sense. One of them is to assume nodes that are oblivious w.r.t. their orientation.

References

1 Greg Aloupis, Nadia Benbernou, Mirela Damian, Erik D. Demaine, Robin Flatland, John Iacono, and Stefanie Wuhrer. Efficient reconfiguration of lattice-based modular robots. *Computational geometry*, 46(8):917–928, 2013.

2 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Computing*, 18(4):235–253, March 2006.

3 Dana Angluin, James Aspnes, David Eisenstat, and Eric Ruppert. The computational power of population protocols. *Distributed Computing*, 20(4):279–304, November 2007.

4 Zack Butler, Keith Kotay, Daniela Rus, and Kohji Tomita. Generic decentralized control for lattice-based self-reconfigurable robots. *The International Journal of Robotics Research*, 23(9):919–937, 2004.

5 Xuli Chen, Li Li, Xuemei Sun, Yanping Liu, Bin Luo, Changchun Wang, Yuping Bao, Hong Xu, and Huisheng Peng. Magnetochromatic polydiacetylene by incorporation of Fe3O4 nanoparticles. *Angewandte Chemie International Edition*, 50(24):5486–5489, 2011.

6 Mark Cieliebak, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Solving the robots gathering problem. In *International Colloquium on Automata, Languages, and Programming*, pages 1181–1196. Springer, 2003.

7 Alejandro Cornejo, Fabian Kuhn, Ruy Ley-Wild, and Nancy Lynch. Keeping mobile robot swarms connected. In *Proceedings of the 23rd international conference on Distributed computing*, DISC'09, pages 496–511, Berlin, Heidelberg, 2009. Springer-Verlag.

8 Shantanu Das, Paola Flocchini, Nicola Santoro, and Masafumi Yamashita. Forming sequences of geometric patterns with oblivious mobile robots. *Distributed Computing*, 28(2):131–145, April 2015.

9 Erik D. Demaine. Playing games with algorithms: Algorithmic combinatorial game theory. In *International Symposium on Mathematical Foundations of Computer Science*, pages 18–33. Springer, 2001.

10 Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Brief announcement: amoebot–a new model for programmable matter. In *Proceedings of the 26th ACM symposium on Parallelism in algorithms and architectures (SPAA)*, pages 220–222, 2014.

11 Zahra Derakhshandeh, Robert Gmyr, Alexandra Porter, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. On the runtime of universal coating for programmable matter. In *International Conference on DNA-Based Computers*, pages 148–164. Springer, 2016.

12 Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. 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*, page 21. ACM, 2015.

13 Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Universal shape formation for programmable matter. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 289–299. ACM, 2016.

14 David Doty. Theory of algorithmic self-assembly. *Communications of the ACM*, 55:78–88, 2012.

15 David Doty. Timing in chemical reaction networks. In *Proc. of the 25th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 772–784, 2014.

16 Adrian Dumitrescu and János Pach. Pushing squares around. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 116–123. ACM, 2004.

17 Adrian Dumitrescu, Ichiro Suzuki, and Masafumi Yamashita. Formations for fast locomotion of metamorphic robotic systems. *The International Journal of Robotics Research*, 23(6):583–593, 2004.

18 Adrian Dumitrescu, Ichiro Suzuki, and Masafumi Yamashita. Motion planning for metamorphic systems: Feasibility, decidability, and distributed reconfiguration. *IEEE Transactions on Robotics and Automation*, 20(3):409–418, 2004.

19 Yuval Emek and Jara Uitto. Dynamic networks of finite state machines. In *International Colloquium on Structural Information and Communication Complexity*, pages 19–34. Springer, 2016.

20 Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Distributed computing by oblivious mobile robots. *Synthesis lectures on distributed computing theory*, 3(2):1–185, 2012.

21 Kyle Gilpin, Ara Knaian, and Daniela Rus. Robot pebbles: One centimeter modules for programmable matter through self-disassembly. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2485–2492. IEEE, 2010.

22 Robert A. Hearn and Erik D. Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science*, 343(1-2):72–96, 2005.

23 Ara N. Knaian, Kenneth C. Cheung, Maxim B. Lobovsky, Asa J. Oines, Peter Schmidt-Neilsen, and Neil A. Gershenfeld. The milli-motein: A self-folding chain of programmable matter with a one centimeter module pitch. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1447–1453. IEEE, 2012.

24 Evangelos Kranakis, Danny Krizanc, and Euripides Markou. The mobile agent rendezvous problem in the ring. *Synthesis Lectures on Distributed Computing Theory*, 1(1):1–122, 2010.

25 Yunfeng Lu, Yi Yang, Alan Sellinger, Mengcheng Lu, Jinman Huang, Hongyou Fan, Raid Haddad, Gabriel Lopez, Alan R. Burns, Darryl Y. Sasaki, John Shelnutt, and C. Jeffrey Brinker. Self-assembly of mesoscopically ordered chromatic polydiacetylene/silica nanocomposites. *Nature*, 410(6831):913–917, 2001.

26 Othon Michail and Paul G. Spirakis. Simple and efficient local codes for distributed stable network construction. *Distributed Computing*, 29(3):207–237, 2016. `doi:http://dx.doi.org/10.1007/s00446-015-0257-4`.

27 Othon Michail and Paul G. Spirakis. Elements of the theory of dynamic networks. *Communications of the ACM*, 2017. Accepted: 6th April 2017, To appear.

28 Paul W. K. Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature*, 440(7082):297–302, 2006.

29 Paul W. K. Rothemund and Erik Winfree. The program-size complexity of self-assembled squares. In *Proceedings of the 32nd annual ACM symposium on Theory of computing (STOC)*, pages 459–468, 2000. `doi:10.1145/335305.335358`.

30 Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.

31 Masahiro Shibata, Toshiya Mega, Fukuhito Ooshita, Hirotsugu Kakugawa, and Toshimitsu Masuzawa. Uniform deployment of mobile agents in asynchronous rings. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, pages 415–424. ACM, 2016.

32 Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM J. Comput.*, 28(4):1347–1363, March 1999. `doi:10.1137/S009753979628292X`.

33 Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, California Institute of Technology, June 1998.

34 Damien Woods, Ho-Lin Chen, Scott Goodfriend, Nadine Dabby, Erik Winfree, and Peng Yin. Active self-assembly of algorithmic shapes and patterns in polylogarithmic time. In

Proceedings of the 4th conference on Innovations in Theoretical Computer Science, pages 353–354. ACM, 2013.

35 Yukiko Yamauchi, Taichi Uehara, and Masafumi Yamashita. Brief announcement: pattern formation problem for synchronous mobile robots in the three dimensional euclidean space. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, pages 447–449. ACM, 2016.

36 Mark Yim, Wei-Min Shen, Behnam Salemi, Daniela Rus, Mark Moll, Hod Lipson, Eric Klavins, and Gregory S. Chirikjian. Modular self-reconfigurable robot systems [grand challenges of robotics]. *IEEE Robotics & Automation Magazine*, 14(1):43–52, 2007.