Work in Progress: An Algorithmic Framework for Shape Formation Problems in Self-Organizing Particle Systems

Zahra Derakhshandeh[*]
Arizona State University, USA
zderakhs@asu.edu

Robert Gmyr[†]
University of Paderborn, Germany
gmyr@mail.upb.de

Andréa W. Richa[*]
Arizona State University, USA
aricha@asu.edu

Christian Scheideler[†]
University of Paderborn, Germany
scheideler@upb.de

Thim Strothmann[†]
University of Paderborn, Germany
thim@mail.upb.de

1. INTRODUCTION

Imagine that we had a piece of matter that can change its physical properties like shape, density, conductivity, or color in a programmable fashion based on either user input or autonomous sensing. This is the vision behind what is commonly known as *programmable matter*. Programmable matter is the subject of many recent novel distributed computing proposals — ranging from DNA tiles, shape-changing molecules, and synthetic cells, to reconfigurable modular robotics — each pursuing solutions for specific application scenarios with their own, special capabilities and constraints.

We envision systems of nano-sensors devices with very limited computational capabilities individually, but which can collaborate to reach a lot more as a collective. Ideally, those nano-sensor devices will be able to self-organize in order to achieve a desired collective goal without the need of central control or external (in particular, human) intervention. For example, one could envision using a system of self-organizing nano-sensor devices to identify and coat leaks on a nuclear reactor or to monitor environmental and structural conditions in abandoned mines, on the exterior of an airplane/spacecraft, or on a bridge, possibly also self-repairing the structure — i.e., realizing what has been coined as "smart paint". The applications in the health arena are also endless, e.g., self-organizing nano-sensor devices could be used within our bodies to detect and coat an area where internal bleeding occurs, eliminating the need of immediate surgery, or they could be used to identify and isolate tumor/malign cells. In many applications, there may be a specific shape that one would like the system to assume (e.g., a disc, or a line, or even an arbitrary convex shape).

Hence, from an algorithmic point-of-view, we are interested in programmable matter consisting of *self-organizing particle systems (SOPS)*, where a *particle* is a simple computational element that can establish and release (communication or physical) bonds and can actively move in a self-organized way, and in general shape formation problems in those systems.

We use the *geometric amoebot model* presented in [2] as our basic model for SOPS: We assume that we have the infinite regular triangular grid graph G_{eqt} that represents the relative positions that a connected set of particles may assume — i.e., the set of nodes in G_{eqt} represents all possible positions of a particle (relative to the other particles in the structure) and each edge represents a possible transition of position (movement) of a particle.

[*]This work was supported in part by the NSF under Awards CCF-1353089 and CCF-1422603.
[†]Supported in part by DFG grant SCHE 1592/3-1

2. OUR CONTRIBUTIONS

In this paper, we present a *general algorithmic framework for shape formation problems* in SOPS, which constitutes of two basic algorithmic primitives: the *spanning forest* primitive and the *snake formation* primitive. We present concrete applications of these primitives to two specific shape formation problems, namely to the problems of having the system of particles self-organize to form the largest possible *hexagonal shape (HEX)* and *triangular shape (TRI)* (defined by a set of inner angles), resp. Both the HEX and TRI formation algorithms are *optimal* with respect to *work*, which we measure by the total number of particle movements needed to reach the desired shape configuration. Our algorithms rely only on local information (e.g., particles do not have ids, nor do they know n, the total number of particles, or have any sort of global coordinate/orientation system), and require only constant-size memory particles.

THEOREM 1. *Our HEX (resp., TRI) algorithm correctly decides HEX (resp., TRI) problem in worst-case optimal $O(n^2)$ work, where n is the number of system particles.*

3. SHAPE FORMATION

We focus on solving *shape formation problems* in the geometric amoebot model starting from any well-initialized connected configuration of particles. We formally define a shape formation problem as a tuple $\mathcal{M} = (\mathcal{I}, \mathcal{G})$ where $\mathcal{I}$ and $\mathcal{G}$ are sets of connected configurations of particles. We say $\mathcal{I}$ is the set of permitted initial configurations and $\mathcal{G}$ is the set of goal configurations. We present a general algorithmic framework for shape formation problems and then specifically investigate the *Hexagonal Shape Formation (HEX)* and the *Triangular Shape Formation (TRI)* problems, where $\mathcal{G}$ would be all configurations of contracted particles where the positions of the set of particles induce the largest possible complete hexagon, or the largest possible complete triangle (given the set of its inner angles), respectively, on G_{eqt}. We say that an algorithm $\mathcal{A}$ *decides* a shape formation problem $\mathcal{M}$, if for any initial configuration from $\mathcal{I}$, all executions of $\mathcal{A}$ eventually reach one of the valid configurations in $\mathcal{G}$ without losing connectivity, and whenever such a system configuration is reached for the first time, the system stays there and all particles decide to perform no further actions.

Before we proceed, we provide some preliminaries. For all algorithms we assume that there is a specific particle we call the *seed* particle, which provides the starting point for constructing the respective shape. If a seed is not available, one can be chosen using the leader election algorithm proposed in [2]. We define the set of *states* that a particle can be in as *inactive*, *follower*, *root*, and *retired*. Initially, all particles are inactive, except the seed particle, which is always in a *retired* state. In addition to its state, each particle p maintains a constant number of (constant-size) *flags*.

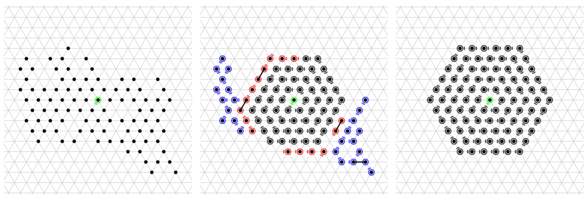
Generally speaking, the shape formation algorithms we propose for hexagonal and triangular shapes progress as follows: Particles organize themselves into a *spanning set of disjoint trees*, where the roots of the trees are non-retired particles adjacent to the partially constructed shape structure (consisting of all retired particles). Root particles lead the way by moving in a predefined direction around the current retired structure. The remaining particles (i.e., the followers) follow behind the leading root particles, making the system "flatten" out towards the direction of movement. Once the leading particles reach a valid position where the shape can be extended (following the rules for the *snake formation* for the particular shape), they stop moving and change their state to retired as well. This process continues until all particles become retired. Once a partcile becomes retired, it performs no further action. Note that the spanning forest component of this general approach is the same for any shape formation algorithm: It is only in the rules that determine the next valid position to be filled in the shape structure being built that the respective algorithms differ. We determine the next valid position to be filled sequentially following a *snake* (i.e., a line of consecutive positions in G_{eqt}), that fills in the space of the respective shape structure and scales naturally with the number of particles in the system. We now briefly explain the general ideas behind the HEX and TRI algorithms.

In order to characterize the snake formation for a given shape formation problem, one only needs to specify the direction in which the line of particles forming the snake should continue to grow, for each new particle added to the snake. Hence once a particle finds the next valid position on the snake, it will become retired and set the snake direction accordingly. Different rules for snake formation will realize different shapes.

Hexagonal Shape.

In the *HEX* problem, the system of particles has to assume the shape of a hexagon (but for the outer layer, which may not be completely full) in G_{eqt}. The hexagon will be built around the seed particle. (A hexagon in G_{eqt} is actually a disk, formed by all nodes of within a certain distance r from the seed node.) We will organize the particles according to a spiral snake structure which will incrementally add new layers to the hexagon, scaling naturally with the number of particles in the system. In a nutshell, the algorithm proceeds as follows: Initially, the seed particle, which is retired, sets the snake direction flag to correspond to one of its adjacent edges. Any particle adjacent to a retired particle becomes a root following the spanning forest algorithm. Each root p moves in a clockwise fashion around the structure of retired particles until it finds the next position to extend the hexagonal snake (i.e., a position connected to a retired particle via an edge marked by the snake direction flag) and becomes retired.

(a)



(b)

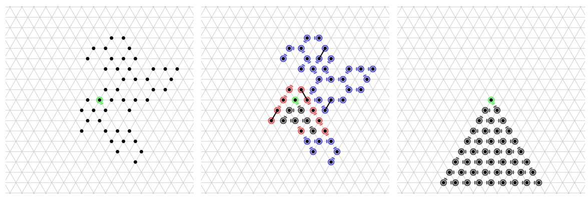


Figure 1: Snapshots of the (a) HEX algorithm and (b) TRI algorithm. The seed is green, retired particles are black, roots are red and followers are blue.

Triangular Shape.

In the *TRI problem*, the system of particles has to assume a final triangular shape on G_{eqt} (but for possibly the outer layer). The snake formation rules for the TRI problem are more complex than the ones we had for the HEX problem, since we will need to explicitly take into account the formation of different layers of particles as we build the triangular structure (this is implicitly taken care of by the spiral formation in the HEX algorithm). The TRI snake construction will start from the seed particle, which will occupy one of the triangle corners. The seed will mark two of its adjacent edges, denote as e_1 and e_2, as the direction along which two of the borders of the triangle will be formed, by setting two special flags on the corresponding edges. These directions will be propagated by the particles that end up on one of the two sides. The seed also marks either e_1 or e_2 as the starting point of forming the triangle snake by setting another special flag, i.e., the snake direction flag. From there on, the triangle snake will be formed layer by layer, alternating going "to the left" and "to the right".

Fig. 1 depicts snapshots of the HEX and TRI algorithms. Detailed descriptions of the algorithms, as well as correctness proofs, appear in [1]; for full simulation runs illustrating both the HEX and TRI algorithms, see http://sops.cs.upb.de.

4. REFERENCES

[1] Z. Derakhshandeh, R. Gmyr, A. W. Richa, C. Scheideler, and T. Strothmann. An algorithmic framework for shape formation problems in self-organizing particle systems. *arXiV*, abs/1504.00744, 2015.

[2] Z. Derakhshandeh, R. Gmyr, T. Strothmann, R. Bazzi, A. W. Richa, and C. Scheideler. Leader election and shape formation with self-organizing programmable matter. To appear in proceedings of DNA21, 2015.