Compaction and Expansion in Self-Organizing Particle Systems

Joshua J. Daymude[1], Miles Laff[1], Zahra Derakhshandeh[1], Robert Gmyr[2], Alexandra Porter[1], Andréa W. Richa[1]

[1] Computer Science, CIDSE, Arizona State University, U.S.A.

[2] Computer Science, University of Paderborn, Germany

Introduction

In the development of *programmable matter*, researchers are tasked with developing synthetic materials that can change their physical properties based on predefined rules and continuous, autonomous collection of input. Often, such programmable matter is highly specific, tailored toward a particular use. In this research, however, we approach this problem from the perspective of theoretical computer science by modeling matter as a system of particles that can perform computations, bond with other particles, and move. Using this geometric "amoebot" model [1], we discuss solutions to the compaction problem—wherein particles form a convex structure containing no holes—and the expansion problem—wherein particles form a ring to enclose spaces.

Amoebot Model

General Assumptions: standard asynchronous model and "compass-free", i.e. there is no global sense of orientation shared amongst the system.

Space: an infinite, undirected graph $G = (V, E)$ in the form of an *equilateral triangular grid* (Figure 1a) in which V is the set of all possible particle positions and E is the set of all possible transitions between positions in V [1, 2].

Particles: constant-size memory, strictly local sense of orientation, and no unique identifiers. Can be either *contracted* or *expanded* (Figure 1b). Can form bonds with their *neighbors* which contain shared, bounded-memory registers used for particle communication.

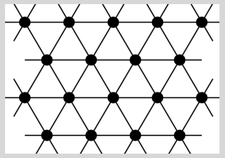

Figure 1a

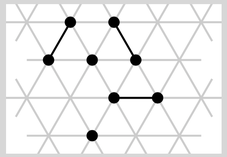

Figure 1b

Movement: achieved via a series of particle *expansions* and *contractions*.

- **Independent:** a contracted particle may expand into an adjacent unoccupied node to become expanded, and completes its movement by contracting to once again occupy only one node.
- **Coordinated:** particles coordinate movements in the form of *handovers*, in which two scenarios are possible: (1) a contracted particle can expand and "push" a neighboring expanded particle, forcing it to contract, or (2) an expanded particle can contract and "pull" a neighboring contracted particle, forcing it to expand. Mimics amoeba movement, and avoids severing the particle system's connectivity [1, 2].

The Compaction Algorithm

A particle is said to be *locally compact* if (1) it does not have exactly five neighbors, and (2) all its neighboring particles occupy consecutive positions. We assume the existence of a single "seed" particle (green) and that the particle system is initially connected. All other particles are initially inactive and without orientation (Figure 2a).

Phase 1: Orientation. Direction is propagated throughout the system by means of spanning trees (Figure 2b).

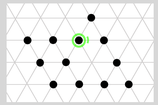

Figure 2a

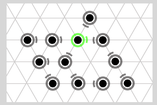

Figure 2b

Phase 2: Movement

- **Type 1: Movement Chains**. If a particle is a parent in its spanning tree and is non-locally compact, it attempts to fix this concavity by moving into an empty position in its neighborhood, becoming a "leader" (Figures 2b & 3a). Its child takes its original place and orientation, becoming a "follower" (Figure 3b). Finally, the rest of the spanning tree follows this chain along its original orientation using handovers (Figures 3c & 3d).

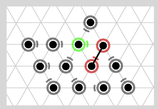

Figure 3a

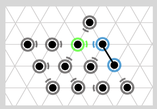

Figure 3b

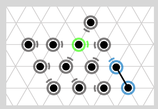

Figure 3c

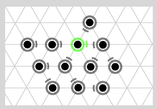

Figure 3d

- **Type 2: Leaf Switching**. A hole bounded entirely by leaves of the system's spanning trees will never be filled by movement chains, as leaves never become leaders. Thus, a leaf not taking part in a movement chain may randomly choose a new parent from its neighbors, guaranteeing the eventual elimination of any hole.

Hole Elimination Algorithms

If one is willing to weaken the definition of compaction by foregoing the requirement of a convex shape, algorithms can focus solely on *hole elimination*. We propose two similar algorithms for hole elimination: *standard* and *compaction-based.* While the two algorithms differ in how particles attempt to gather together, they share the following local rules for when a particle may finish:

(1) a particle may finish if it occupies a position which lies radially outward from the seed; otherwise,

(2) a particle may finish if it has three consecutive finished particles in its neighborhood.

Hole Elimination Algorithms (cont.)

Phase 1: Orientation. Direction is propagated throughout the system just as it is for the compaction algorithm.

Phase 2: Gathering. Members of spanning trees attempt to get closer to their root.

- **Standard**. In the standard hole elimination algorithm, members of spanning trees are only allowed to move by performing handovers with their parent in the tree, essentially following a fixed path defined by the tree's structure.
- **Compaction-Based**. In the compaction-based version, members of spanning trees behave as they do in the compaction algorithm's "Movement" phase.

Phase 3: Border Walking. A particle comes into contact with some finished particle and proceeds to walk counterclockwise around the finished border until discovering a position where it also may become finished.

Hexagon Shape Formation

Our particle systems support another class of algorithms which achieve "shape formation" of various kinds. In the case of *hexagon shape formation,* the particle system assembles a structure that is as close to a sphere as possible and contains no holes; thus, it is (nearly) compact. This is achieved using a "snake-like" approach similar to the border walking found in standard hole elimination.

Performance Comparisons

We were interested in comparing the different compaction and hole elimination algorithms against one another. In theory, we've shown that both standard hole elimination and hexagon shape formation require $\Theta(n^2)$ work, as does compaction-based hole elimination on expectation. Furthermore, since the hole elimination algorithms provide more opportunities for particles to finish, we expected them to terminate in less movements. However, as Figure 4 shows, hexagon shape formation (blue) consistently outperforms both hole elimination algorithms (orange and green) when run on the same initial configurations.

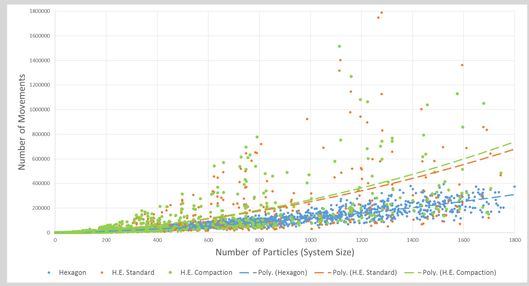

Figure 4

The Expansion Algorithm

When a particle system wishes to surround an area or structure, a natural movement to perform is to spread out over as large an area as possible; i.e. form a spherical shell. In two dimensions, a particle system is said to be *expanded* if it forms a ring [3], that is, the 2-connected structure with the smallest number of edges. Assumptions of initial connectivity and the existence of a seed hold as in the compaction algorithm.

Phase 1: Seed Surrounding. By surrounding the seed, an initial ring configuration is established.

Phase 2: Line Formation. Once the seed is surrounded, one of its neighbors points opposite the seed to begin the line. When another particle moves into the position incident to this point, it stops moving and likewise points in the direction opposite the seed. Thus, a line (Figure 5, depicted in yellow) grows outwards from the seed.

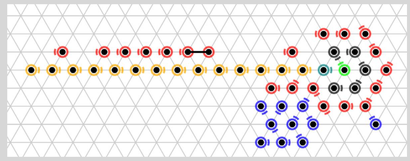

Figure 5

Phase 3: Line Collapsing. The previously formed line collapses to either side of itself to form the sides of the ring. Each complete collapsing movement increases the area of the ring by one, and these movements continue until the line is exhausted, completing the ring.

Conclusion

The general amoebot model provides simple, robust means of describing programmable matter. By abstracting above physical implementation, the algorithms provided for *compaction* and *expansion* are more widely applicable in different physical systems. The proposed compaction algorithms explore different ways of balancing the tension between algorithm efficiency and shape of the final structure; similarly, future approaches to expansion must also attempt to achieve more optimally expanded structures without sacrificing runtime.

References

1. Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Brief announcement: amoebot - a new model for programmable matter. In *26th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '14)*, pages 220–222. ACM, 2014.
2. Zahra Derakhshandeh, Robert Gmyr, Thim Strothmann, Rida A. Bazzi, Andréa W. Richa, and Christian Scheideler. Leader election and shape formation with self-organizing programmable matter. 2015. To appear, CoRR abs/1503.07991.
3. Miles Laff. Expansion algorithms in self-organizing particle systems. Honors Bachelor's thesis, Barrett Honors College, Arizona State University, 2015.

TEMPLATE DESIGN © 2007
www.PosterPresentations.com