Brief Announcement: Fault-Tolerant Shape Formation in the Amoebot Model

Irina Kostitsyna
Department of Mathematics and Computer Science,
Eindhoven University of Technology, The Netherlands

Christian Scheideler
Department of Computer Science, Paderborn University, Germany

Daniel Warner
Department of Computer Science, Paderborn University, Germany

Abstract

The amoebot model is a distributed computing model of programmable matter. It envisions programmable matter as a collection of computational units called amoebots or particles that utilize local interactions to achieve tasks of coordination, movement and conformation. In the geometric amoebot model the particles operate on a hexagonal tessellation of the plane. Within this model, numerous problems such as leader election, shape formation or object coating have been studied. One area that has not received much attention so far, but is highly relevant for a practical implementation of programmable matter, is fault tolerance. The existing literature on that aspect allows particles to crash but assumes that crashed particles do not recover. We propose a new model in which a crash causes the memory of a particle to be reset and a crashed particle can detect that it has crashed and try to recover using its local information and communication capabilities. We propose an algorithm that solves the hexagon shape formation problem in our model if a finite number of crashes occur and a designated leader particle does not fail. At the heart of our solution lies a fault-tolerant implementation of the spanning forest primitive, which, since other algorithms in the amoebot model also make use of it, is also of general interest.

2012 ACM Subject Classification General and reference → General conference proceedings

Keywords and phrases Programmable matter, Geometric amoebot model, Fault tolerance, Shape formation

Digital Object Identifier 10.4230/LIPIcs.SAND.2022.23

1 Introduction

Model extension

In our work we extend the amoebot model by introducing *particle crashes*. In order to gain initial insights into useful strategies towards fault tolerance in our model and motivate further work in this direction, we focus on the problem of shape formation in the geometric amoebot model using the hexagon shape formation problem as basis. We assume that the adversarial scheduler may arbitrarily crash particles. A crash of a particle p has the following effects: The scheduler sets the *state* in p's local memory to CRASHED, enabling p and its neighbours to detect that it has crashed, and resets the rest of p's local memory. The faulty particle p can then try to recover its local memory by using its local information and communication capabilities.

Problem description

For any two nodes $u, v \in V_\triangle$ of the triangular lattice $G_\triangle$ the *distance* $\delta(u,v) \in \mathbb{N}_0$ between u and v is defined as the length of a shortest path from u to v in $G_\triangle$. For a node $v \in V_\triangle$ and $i \in \mathbb{N}_0$ let $B(v,i) := \{\, u \in V_\triangle \mid \delta(u,v) = i \,\}$. We call a set $V \subseteq V_\triangle$ a *hexagon with centre* $v \in V$ if there is a $k \in \mathbb{N}_0$ and a subset $S \subseteq B(v,k)$ such that $V = S \cup \bigcup_{i<k} B(v,i)$. We

© Irina Kostitsyna, Christian Scheideler, and Daniel Warner;
licensed under Creative Commons License CC-BY 4.0

1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2022).
Editors: James Aspnes and Othon Michail; Article No. 23; pp. 23:1–23:3
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

define the *hexagon shape formation problem* **HEX**: We assume that the system of particles initially forms a single connected component of contracted particles, has a unique leader, called the SEED particle, and that all other particles are IDLE. The goal is to reach a stable configuration in which the set of nodes occupied by particles is a hexagon with the SEED in its centre.

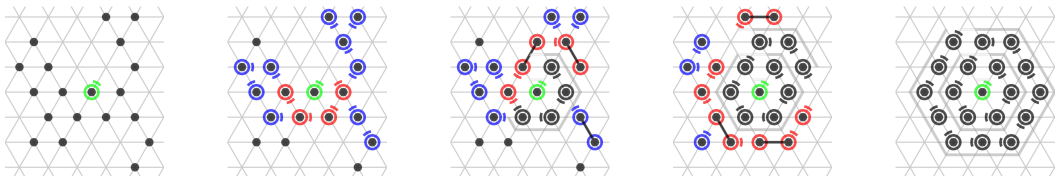
2 Main results

We propose an algorithm `HexagonFT` that solves the hexagon shape formation problem **HEX** in our model under the presence of particle crashes. Our two main results are:

▶ **Lemma 1.** *If a finite number of crashes occur during the execution of algorithm* `HexagonFT` *and m particles are faulty after the last crash, then a non-faulty configuration is reached within $\mathcal{O}(mn)$ rounds after the last crash.*

▶ **Theorem 2.** *If a finite number of crashes occur, then the algorithm* `HexagonFT` *solves the hexagon shape formation problem* **HEX** *in worst-case $\mathcal{O}(n^2)$ work (total number of moves executed by all particles). From the time when no more crashes occur and the configuration is non-faulty, the algorithm needs $\mathcal{O}(n)$ rounds until termination.*

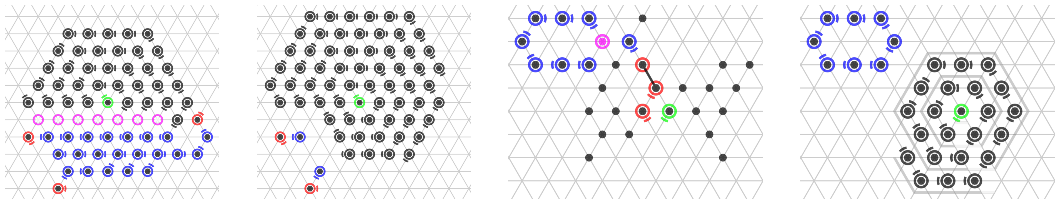
As long as no crashes occur, `HexagonFT` behaves like the classical hexagon shape formation algorithm introduced in [1] (compare Figure 1).



(a) (b) (c) (d) (e)

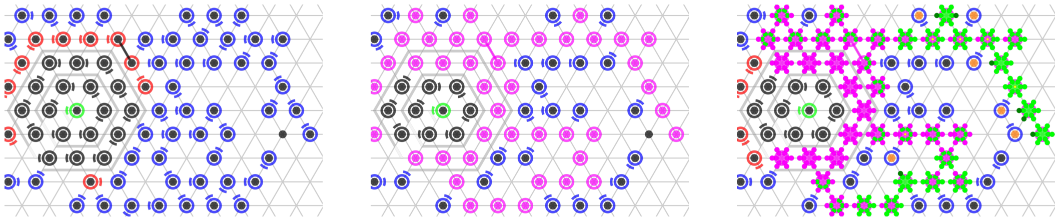
Figure 1 An example run of our hexagon shape formation algorithm `HexagonFT` with 19 particles without crashes. Particles have to assume the shape of a hexagon (but for the outer layer, which may not be completely full). The hexagon is built in a spiral ring in clockwise direction around the SEED as follows: (a) All particles except of the SEED are initially IDLE (black dots). (b) Particles adjacent to *finished* particles (SEED or RETIRED) become ROOT particles, and FOLLOWER particles form parent-child relationships with ROOT or FOLLOWER particles. (c)–(e) ROOT particles traverse the forming hexagon counter-clockwise, becoming RETIRED when reaching the position marked by the last RETIRED particle. FOLLOWER particles follow ROOT particles via a series of handovers.

Due to space limitations, we address two algorithmic challenges that arise due to particle crashes (Figure 2): Firstly, we must ensure that particles within the hexagon formed so far do not become FOLLOWER particles. If this is not ensured, particles could leave the hexagon, which in turn could lead to disconnection of the particles. We use a *safety primitive* (Figure 3) to ensure that particles inside the hexagon cannot become FOLLOWER particles. Secondly, we need to ensure that when a crashed particle chooses a FOLLOWER as parent, this does not lead to disconnection of the particles. In order to avoid disconnection, we use a *validation primitive* (Figure 4) that determines for a faulty particle which of the FOLLOWER parent candidates it can attach to without closing a cycle.



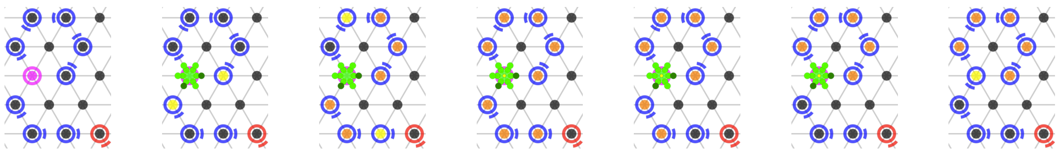
(a) (b) (c) (d)

Figure 2 (a)–(b) Crashed particles inside the hexagon have become followers. Some of these followers follow their root, causing them to leave the hexagon, which eventually leads to a disconnection of the particles. (c)–(d) A Crashed particle attaches itself to an arbitrary FOLLOWER pointing away from it, closing a cycle and leading to irreversible disconnection of the particles.



(a) (b) (c)

Figure 3 *Safety primitive*: Crashed particles will become either SAFE or UNSAFE. Crashed particles connected to a finished particle via one or two line segments in $G_\triangle$ become UNSAFE, otherwise SAFE by the propagation of *safeFlags*. Only a SAFE particle may become a FOLLOWER.



(a) (b) (c) (d) (e) (f) (g)

Figure 4 *Validation primitive*: (a) A faulty particle needs to ensure that there is a path to a ROOT before following a particle. (b) The faulty particle sends INVALIDATE tokens to possible parent candidates. (c) INVALIDATE is propagated upwards, causing particles on the path to become INVALID. (d)–(e) An INVALIDATE that reaches an ERROR particle is stored by it, an INVALIDATE that reaches a ROOT is consumed by it. The ROOT generates a VALID token which is propagated downwards along the INVALID particles, causing them to become VALID. (f)–(g) A SAFE particle may become a FOLLOWER of a VALID FOLLOWER parent candidate. After the recovery of the particle, the INVALIDATE token previously stored by it will then again be propagated upwards.

References

1 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*, pages 1–2, 2015.