Reconfiguration and Locomotion with Joint Movements in the Amoebot Model*

Andreas Padalkin[1], Manish Kumar[2], and Christian Scheideler[3]

1 **Paderborn University, Germany**
andreas.padalkin@upb.de
2 **Bar-Ilan University, Israel**
manish.kumar@biu.ac.il
3 **Paderborn University, Germany**
scheideler@upb.de

Abstract

We are considering the geometric amoebot model where a set of n *amoebots* is placed on the triangular grid. An amoebot is able to send information to its neighbors, and to move via expansions and contractions. Since amoebots and information can only travel node by node, most problems relevant for programmable matter have a natural lower bound of $\Omega(D)$ where D denotes the diameter of the structure. Inspired by the nervous and muscular system, Feldmann et al. have proposed the *reconfigurable circuit extension* and the *joint movement extension* of the amoebot model with the goal of breaking this lower bound.

In the joint movement extension, the way amoebots move is altered. Amoebots become able to push and pull other amoebots. Feldmann et al. demonstrated the power of joint movements by transforming a line of amoebots into a rhombus within $O(\log n)$ rounds. However, they left the details of the extension open. The goal of this paper is therefore to formalize the joint movement extension. In order to provide a proof of concept for the extension, we consider two fundamental problems of modular robot systems: *reconfiguration* and *locomotion*.

We approach these problems by defining meta-modules of rhombical and hexagonal shape, respectively. The meta-modules are capable of movement primitives like sliding, rotating, and tunneling. This allows us to simulate reconfiguration algorithms of various modular robot systems. Finally, we construct three amoebot structures capable of locomotion by rolling, crawling, and walking, respectively.

Related Version *Full Version*: https://arxiv.org/abs/2305.06146

1 Introduction

Programmable matter consists of homogeneous nano-robots that are able to change the properties of the matter in a programmable fashion, e.g., the shape, the color, or the density [18]. We are considering the geometric amoebot model [3, 4, 5] where a set of n nano-robots called *amoebots* is placed on the triangular grid. An amoebot is able to send information to its neighbors, and to move by first expanding into an unoccupied adjacent node, and then contracting into that node. Since amoebots and information can only travel node by node, most problems relevant for programmable matter have a natural lower bound of $\Omega(D)$ where D denotes the diameter of the structure. Inspired by the *nervous* and *muscular system*, Feldmann et al. [9] proposed the *reconfigurable circuit extension* and the *joint movement extension* with the goal of breaking this lower bound.

* This work has been supported by the DFG Project SCHE 1592/10-1 and Israel Science Foundation under Grants 867/19 and 554/23.

40th European Workshop on Computational Geometry, Ioannina, Greece, March 13–15, 2024.
This is an extended abstract of a presentation given at EuroCG'24. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

In the reconfigurable circuit extension, the amoebot structure is able to interconnect amoebots by *circuits*. Each amoebot can send a primitive signal on circuits it is connected to. The signal is received by all amoebots connected to the same circuit. Among others, Feldmann et al. [9] solved the leader election problem, compass alignment problem, and chirality agreement problem within $O(\log n)$ rounds with high probability (w.h.p.). These problems will be important to coordinate the joint movements. Afterwards, Padalkin et al. [16] explored the structural power of the circuits by considering the spanning tree problem and symmetry detection problem. Both problems can be solved within polylogarithmic time w.h.p.

In the joint movement extension, the way amoebots move is altered. In a nutshell, an expanding amoebot is capable of pushing other amoebots away from it, and a contracting amoebot is capable of pulling other amoebots towards it. Feldmann et al. [9] demonstrated the power of joint movements by transforming a line of amoebots into a rhombus within $O(\log n)$ rounds. However, they left the details of the extension open. The goal of this paper is therefore to formalize the joint movement extension. In order to provide a proof of concept for the extension, we consider two fundamental problems of modular robot systems (MRS): reconfiguration and locomotion. We study these problems from a centralized view to explore the limits of the extension.

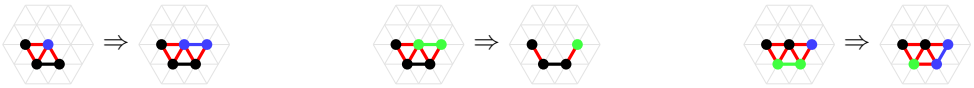
In the reconfiguration problem, an MRS has to reconfigure its structure into a given shape. Examples for reconfiguration algorithms in the original amoebot model can be found in [6, 7, 14, 15]. However, all of these are subject of the aforementioned natural lower bound. To our knowledge, polylogarithmic time solutions were found for two types of MRSs: in the nubot model [20] and crystalline atom model [2].

In the locomotion problem, an MRS has to move along an even surface as fast as possible. In the original amoebot model, one would use the spanning tree primitive to move along the surface [3]. However, we only obtain a constant velocity with that. In terrestrial environments, there are three basic types of locomotion: rolling, crawling, and walking [11, 13]. For each of these locomotion types, we will present an amoebot structure.

2 Geometric Amoebot Model

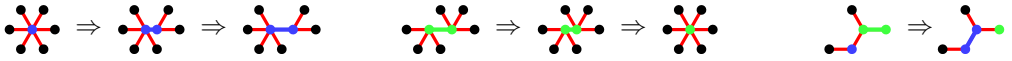
In this section, we introduce the *geometric amoebot model* [4]. We slightly deviate from the original model to make it more suitable to our extension. A set of n amoebots is placed on the infinite regular triangular grid graph $G_\Delta = (V, E)$ (see Figure 1). An amoebot is an anonymous, randomized finite state machine in form of a line segment. The endpoints may either occupy the same node or two adjacent nodes. If the endpoints occupy the same node, the amoebot has length 0 and is called *contracted* and otherwise, it has length 1 and is called *expanded*. Every node of G_Δ is occupied by at most one amoebot. Two endpoints of different amoebots that occupy adjacent nodes in G_Δ are connected by *bonds* (red edges). An amoebot can move through *contractions* and *expansions*. We refer to [4] for more details.

Let the *amoebot structure* $S \subseteq V$ be the set of nodes occupied by the amoebots. We say



(a) Expansion. **(b)** Contraction. **(c)** Handover.

Figure 1 Movement in the geometric amoebot model. Red lines indicate bonds. Blue amoebots are expanding. Green amoebots are contracting.



(a) Expansion. **(b)** Contraction. **(c)** Handover.

Figure 2 Movements in the extension. Red lines indicate bonds. Blue amoebots are expanding. Green amoebots are contracting. The first two figures show a movement in 0.5 time steps.

that S is *connected* if and only if G_S is connected, where $G_S = G_\Delta|_S$ is the graph induced by S. Initially, S is connected. Also, we assume the fully synchronous activation model, i.e., the time is divided into synchronous rounds, and every amoebot is active in each round. We justify this assumption with the reconfigurable circuit extension. On activation, each amoebot may perform a movement and update its state as a function of its previous state. However, if an amoebot fails to perform its movement, it remains in its previous state. The time complexity of an algorithm is measured by the number of synchronized rounds required by it.

3 Joint Movement Extension

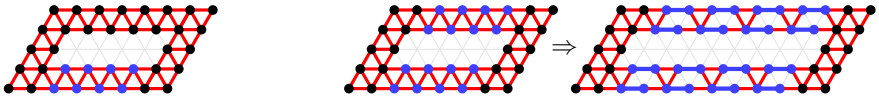
In the *joint movement extension* [9], the way the amoebots move is altered. The idea behind the extension is to allow amoebots to push and pull other amoebots. The necessary coordination of such movements can be provided by the reconfigurable circuit extension [9, 16]. In the following, we formalize the joint movement extension. Joint movements are performed in two steps.

In the first step, the amoebots remove bonds from G_S as follows. Each amoebot can decide to release an arbitrary subset of its currently incident bonds in G_S. A bond is removed if and only if either of the amoebots at the endpoints releases the bond. Let $E_L \subseteq E_S$ denote the set of all edges occupied by amoebots and $E_R \subseteq E_S$ the set of the remaining bonds, and $G_R = (S, E_L \cup E_R)$ the resulting graph. We require that G_R is connected since otherwise, disconnected parts might float apart. We say that a *connectivity conflict* occurs if and only if G_R is not connected. Whenever a connectivity conflict occurs, the amoebot structure transitions into an undefined state such that we become unable to make any statements about the structure.

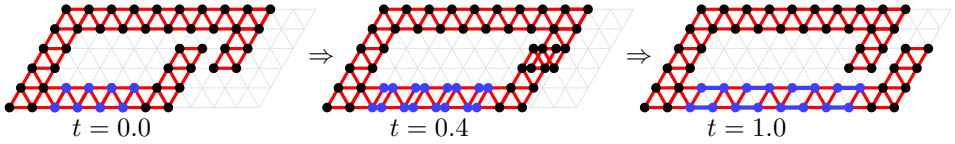
In the second step, each amoebot may perform one of the following movements within the time period $[0, 1]$. A contracted amoebot may expand on one of the axes as follows (see Figure 2a). At $t = 0$, the amoebot can reorientate itself and reassign each of its incident bonds to one of its endpoints. At $t \in [0, 1]$, the amoebots has a length of t. In the process, the incident bonds do not change their orientations or lengths. As a result, the expanding amoebot pushes all connected amoebots. An expanded amoebot may contract analogously by reversing the contraction (see Figure 2b). Thereby, it pulls all connected amoebots.

Furthermore, a contracted amoebot x occupying node u and an expanded amoebot y occupying nodes v and w may perform a handover if there is a bond b between u and v, as follows (see Figure 2c). At an arbitrary $t \in [0, 1]$, we flip $b = \{u, v\}$ and $\{v, w\}$ such that x becomes an expanded amoebot with endpoints occupying nodes u and v, y becomes a contracted amoebot with both endpoints occupying w, and b becomes $\{v, w\}$. We have to include the handover to ensure universality of the model since otherwise, it would not be possible to move through a narrow tunnel.

In certain situations, the amoebots may not be able to perform their movements. We distinguish between two cases. First, the amoebots may not be able to perform their movements while maintaining their relative positions (see Figure 3a). We call that a *structural*



(a) Structural conflicts and mappings. In the left figure, the expansions cause a structural conflict. In the right figure, the amoebot structure can be mapped onto the triangular grid after the movement.



(b) Collision. Initially, we have a valid amoebot structure given ($t = 0$). The resulting amoebot structure can be mapped on G_Δ ($t = 1$). However, for $t \in [0.25, 0.75]$, parts of the structure collide.

Figure 3 Joint movements. Red lines indicate bonds. Blue amoebots are expanding horizontally.

conflict. Second, parts of the structure may collide into each other. More precisely, a *collision* occurs if there is a $t \in [0, 1]$ such that two non-adjacent bonds intersect at some point (see Figure 3b). Whenever either a structural conflict or a collision occurs, the amoebot structure transitions into an undefined state such that we become unable to make any statements about the structure. The detection of structural conflicts and collisions is not within the scope of this paper simply because we only consider movements where structural conflicts and collisions cannot occur. We refer to [10] for more details.

Otherwise, at $t = 1$, we obtain a graph $G_M = (S, E_M)$ that can be mapped on the triangular grid G_Δ (see Figure 3a). In compliance with the orientations of all bonds and line segments, the mapping of G_M is unique except for translations since G_R is connected. We choose any mapping as our next amoebot structure. Afterwards, the amoebots reestablish all possible bonds.

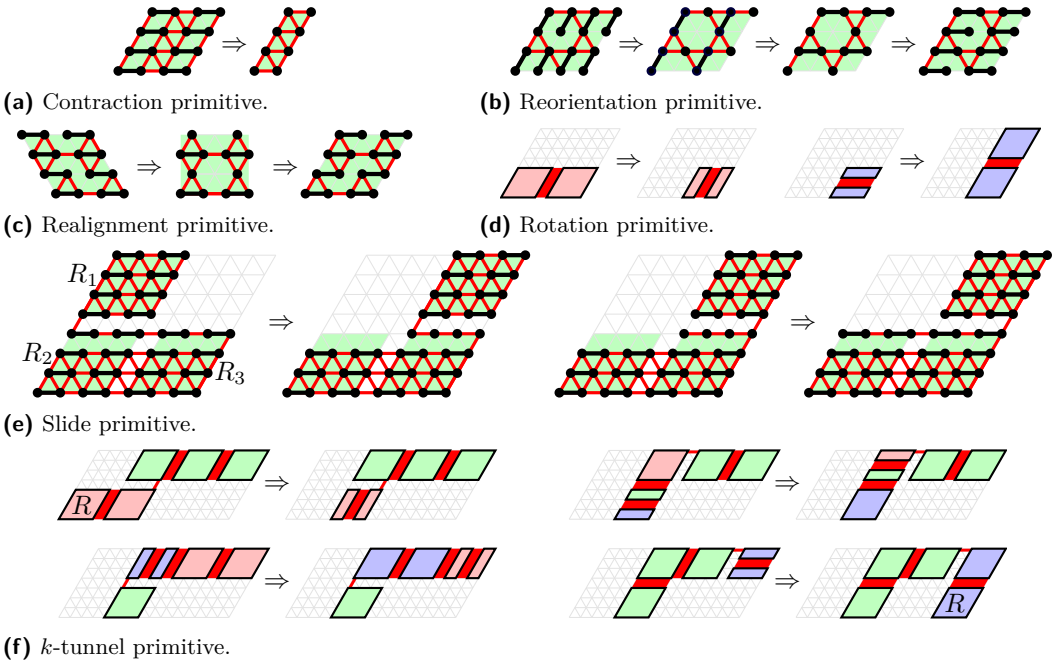
We assume that the joint movements are performed within *look-compute-move* cycles. In the *look phase*, each amoebot observes its neighborhood and receives signals (beeps) from other amoebots, according to the reconfigurable circuit structure of the system. In the *compute phase*, each amoebot may perform computations, change its state, and decide the actions to perform in the next phase (i.e., which bonds to release, and which movement to perform). In the *move phase*, each amoebot may release an arbitrary subset of its incident bonds, and perform a movement.

4 Proof of Concept

We now provide a proof of concept for the joint movement extension by considering two fundamental problems: reconfiguration and locomotion.

In a first step, we combine multiple amoebots to meta-modules. In other models for programmable matter and modular robots, meta-modules have proven to be very useful. For example, they allow us to bypass restrictions on the reconfigurability [8, 19] and to simulate (reconfiguration) algorithms for other models [1, 17]. We present two types of meta-modules: meta-modules of rhombical and hexagonal shape. For these, we can show various movement primitives (see Figures 4 and 5).

The meta-modules allow us to simulate reconfiguration algorithms for lattice-type MRSs of similar shape if we can implement the same movement primitives. By simulating the reconfiguration algorithm of Aloupis et al. [2] for rhombical robots (crystalline atoms) and the reconfiguration algorithm of Hurtado et al. [12] for hexagonal robots, we obtain the



(a) Contraction primitive.

(b) Reorientation primitive.

(c) Realignment primitive.

(d) Rotation primitive.

(e) Slide primitive.

(f) k-tunnel primitive.

Figure 4 Movement primitives for rhombical meta-modules. Red meta-modules perform a pull operation, and blue meta-modules a push operation.

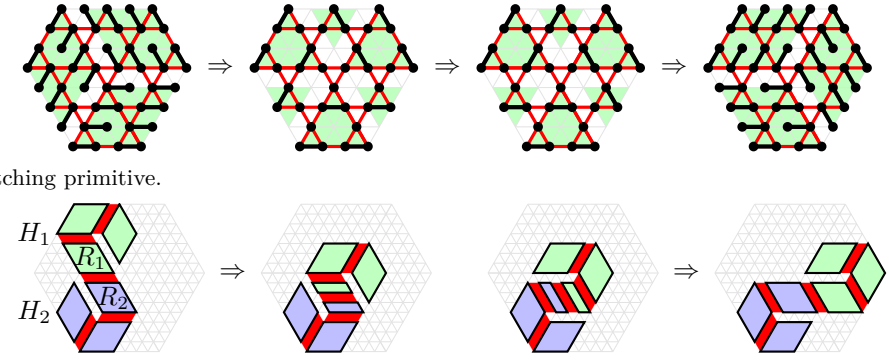
following results.

▶ **Theorem 4.1.** *There is a centralized reconfiguration algorithm for m hexagonal meta-modules that requires $O(m)$ rounds. Each module has to perform at most $O(m)$ moves.*

▶ **Theorem 4.2.** *There is a centralized reconfiguration algorithm for m rhombical meta-modules that requires $O(\log m)$ rounds and performs $\Theta(m \log m)$ moves overall.*

Finally, we consider amoebot structures capable of locomotion along an even surface. There are three basic types of terrestrial locomotion: rolling, crawling, and walking [11, 13]. In the following, we will construct an amoebot structure for each type.

Our rolling amoebot structure imitates a continuous track that rotates around a set of wheels. We build it from hexagonal meta-modules of alternating side lengths ℓ and $\ell - 1$



(a) Switching primitive.

(b) Rotation primitive.

Figure 5 Movement primitives for hexagonal meta-modules.

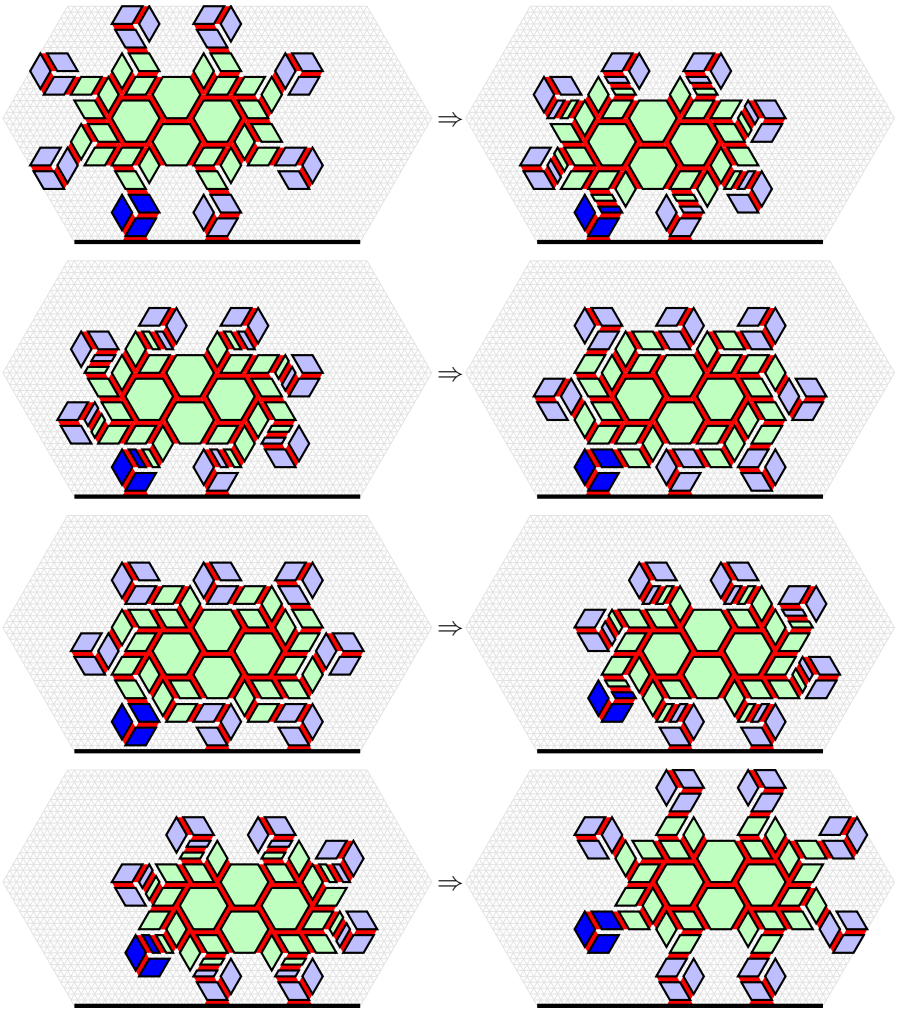


Figure 6 Rolling structure. The blue meta-modules rotate clockwise around the green meta-modules. We highlight one of the rotating meta-modules in a darker blue.

(see Figure 6). The structure consists of two parts: a connected substrate structure (green meta-modules), and a closed chain of meta-modules rotating along the outer boundary of the substrate (blue meta-modules). The amoebot structure moves by rotating the blue meta-modules around the substrate (compare to Figure 5b). We obtain the initial structure after two rotations. In doing so, the structure has moved a distance of $2 \cdot \ell$. By performing the movements periodically, we obtain the following theorem.

▶ **Theorem 4.3.** *Our rolling structure composed of hexagonal meta-modules of alternating side lengths ℓ and $\ell - 1$ moves a distance of $2 \cdot \ell$ within each period of constant length.*

Our crawling amoebot structure imitates earthworms. It consists of r rhombical meta-modules of side length $\ell - 1$ (see Figure 7). The amoebot structure moves by alternately contacting and expanding its body (compare to Figure 4a) while utilizing the meta-module at the front and at the end as an anchor, respectively. As a result, the contraction (expansion) pulls (pushes) the structure to the front. By performing the movements periodically, we obtain the following theorem.

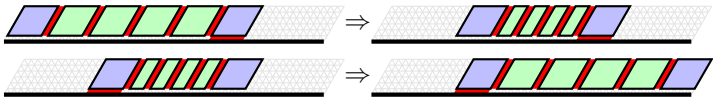


Figure 7 Crawling structure.

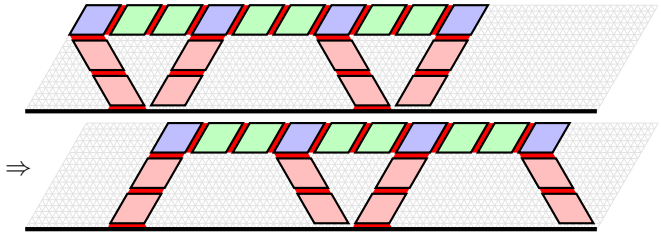


Figure 8 Walking structure.

▶ **Theorem 4.4.** *A line of r rhombical meta-modules of side length $\ell - 1$ moves a distance of $\frac{r-2}{2} \cdot \ell$ every 2 rounds.*

Our walking amoebot structure imitates millipedes. It consists of rhombical meta-modules of side length $\ell - 1$ (compare to Figure 8). Let p denote the number of legs. The body and each leg consists of a line of q rhombical meta-modules. The structure moves by moving the legs back and forth. For that, we simply apply the realignment primitive (see Figure 4c) on all meta-modules within the legs. Note that we reach the initial amoebot structure after two leg movements. Hence, we obtain the following theorem.

▶ **Theorem 4.5.** *Our walking structure composed of rhombical meta-modules of side length $\ell - 1$ with p legs composed of q rhombical meta-modules moves a distance of $2 \cdot q \cdot \ell$ within each period of constant length.*

References

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

2 Greg Aloupis, Sébastien Collette, Erik D. Demaine, Stefan Langerman, Vera Sacristán Adinolfi, and Stefanie Wuhrer. Reconfiguration of cube-style modular robots using O(log n) parallel moves. In *ISAAC*, volume 5369 of *Lecture Notes in Computer Science*, pages 342–353. Springer, 2008.

3 Joshua J. Daymude, Kristian Hinnenthal, Andréa W. Richa, and Christian Scheideler. Computing by programmable particles. In *Distributed Computing by Mobile Entities*, volume 11340 of *Lecture Notes in Computer Science*, pages 615–681. Springer, 2019.

4 Joshua J. Daymude, Andréa W. Richa, and Christian Scheideler. The canonical amoebot model: Algorithms and concurrency control. In *DISC*, volume 209 of *LIPIcs*, pages 20:1–20:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

5 Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Brief announcement: amoebot - a new model for programmable matter. In *SPAA*, pages 220–222. ACM, 2014.

6 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 *NANOCOM*, pages 21:1–21:2. ACM, 2015.

7 Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Universal shape formation for programmable matter. In *SPAA*, pages 289–299. ACM, 2016.

8 Daniel J. Dewey, Michael P. Ashley-Rollman, Michael DeRosa, Seth Copen Goldstein, Todd C. Mowry, Siddhartha S. Srinivasa, Padmanabhan Pillai, and Jason Campbell. Generalizing metamodules to simplify planning in modular robotic systems. In *IROS*, pages 1338–1345. IEEE, 2008.

9 Michael Feldmann, Andreas Padalkin, Christian Scheideler, and Shlomi Dolev. Coordinating amoebots via reconfigurable circuits. *J. Comput. Biol.*, 29(4):317–343, 2022.

10 Siddharth Gupta, Marc J. van Kreveld, Othon Michail, and Andreas Padalkin. Collision detection for modular robots - it is easy to cause collisions and hard to avoid them. *CoRR*, abs/2305.01015, 2023.

11 Shigeo Hirose. Three basic types of locomotion in mobile robots. In *Fifth International Conference on Advanced Robotics' Robots in Unstructured Environments*, pages 12–17. IEEE, 1991.

12 Ferran Hurtado, Enrique Molina, Suneeta Ramaswami, and Vera Sacristán Adinolfi. Distributed reconfiguration of 2d lattice-based modular robotic systems. *Auton. Robots*, 38(4):383–413, 2015.

13 Matthew Kehoe and Davide Piovesan. Taxonomy of two dimensional bio-inspired locomotion systems. In *EMBC*, pages 3703–3706. IEEE, 2019.

14 Irina Kostitsyna, Christian Scheideler, and Daniel Warner. Fault-tolerant shape formation in the amoebot model. In *DNA*, volume 238 of *LIPIcs*, pages 9:1–9:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

15 Giuseppe Antonio Di Luna, Paola Flocchini, Nicola Santoro, Giovanni Viglietta, and Yukiko Yamauchi. Shape formation by programmable particles. *Distributed Comput.*, 33(1):69–101, 2020.

16 Andreas Padalkin, Christian Scheideler, and Daniel Warner. The structural power of reconfigurable circuits in the amoebot model. In *DNA*, volume 238 of *LIPIcs*, pages 8:1–8:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

17 Irene Parada, Vera Sacristán, and Rodrigo I. Silveira. A new meta-module for efficient reconfiguration of hinged-units modular robots. In *ICRA*, pages 5197–5202. IEEE, 2016.

18 Tommaso Toffoli and Norman Margolus. Programmable matter: Concepts and realization. *Int. J. High Speed Comput.*, 5(2):155–170, 1993.

19 Sergei Vassilvitskii, Jeremy Kubica, Eleanor Gilbert Rieffel, John W. Suh, and Mark Yim. On the general reconfiguration problem for expanding cube style modular robots. In *ICRA*, pages 801–808. IEEE, 2002.

20 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 *ITCS*, pages 353–354. ACM, 2013.