Silent Programmable Matter: Coating

Alfredo Navarra
Department of Mathematics and Computer Science, University of Perugia, Italy

Francesco Piselli[1]
Department of Mathematics and Computer Science, University of Perugia, Italy

Abstract

By *Programmable Matter* (PM) is usually meant a system of weak and self-organizing computational entities, called *particles*, which can be programmed via distributed algorithms to collectively achieve some global tasks. We consider the SILBOT model where particles are modeled as finite state automata, living and operating in the cells of a hexagonal grid. Particles are all identical, executing the same deterministic algorithm which is based on local observation of the surroundings, up to two hops. Particles are asynchronous, without any direct means of communication and disoriented but sharing a common handedness, i.e., *chirality* is assumed. Within such a basic model, we consider a foundational primitive for PM, that is *Coating*: a set of n particles must move so as to ensure the closed surrounding of an object occupying some connected cells of the grid. We present an optimal deterministic distributed algorithm – along with the correctness proof, that in $\Theta(n^2)$ rounds solves the Coating problem, where a round concerns the minimal time window within which each particle is activated at least once.

2012 ACM Subject Classification Theory of computation → Distributed algorithms; Theory of computation → Concurrency; Theory of computation → Self-organization

Keywords and phrases Programmable Matter, Coating, Asynchrony, Stigmergy

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2023.25

Funding This work has been funded by the European Union – NextGenerationEU under the Italian Ministry of University and Research (MUR) National Innovation Ecosystem grant ECS00000041 – VITALITY – CUP J97G22000170005, and by the Italian National Group for Scientific Computation (GNCS-INdAM).

1 Introduction

In distributed computing, the research for a basic model to address different situations is certainly one of the mostly investigated issues. On the one hand there is reality, with powerful entities like agents or robots that provide many useful capabilities. On the other hand, models should be formalized with the minimal set of assumptions. In fact, the weaker a model, the wider the range of applications, as well as dependability and robustness with respect to possible disruptions increase. Consequently, also the algorithms designed for weak models turn out to cover a wider set of possible scenarios. Within this context, we investigate on the so-called Programmable Matter (PM). This is intended as a matter with the ability to change its physical properties (e.g., shape, optical properties, etc.) in a programmable way. From its origin [36], PM denotes a system of computational entities, called *particles*, that can be programmed via distributed algorithms to collectively achieve global tasks. Initially, such systems have found main interest in the field of biology and nano-technologies, see e.g., [14, 34]. However, more and more natural applications in various contexts can be imagined, including smart materials, ubiquitous computing, repairing at microscopic scale, and tools for minimally invasive surgery.

[1] Corresponding author.

27th International Conference on Principles of Distributed Systems (OPODIS 2023).
Editors: Alysson Bessani, Xavier Défago, Junya Nakamura, Koichi Wada, and Yukiko Yamauchi; Article No. 25; pp. 25:1–25:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

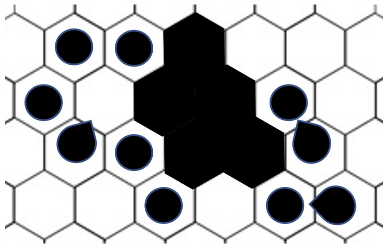

Figure 1 A configuration in SILBOT with an object represented by five connected full-black cells, seven CONTRACTED particles represented by black circles and three EXPANDED particles represented by black drop-like shapes expressing the intent to move toward a neighboring cell.

So far, one of the most investigated models for PM is certainly the geometric *Amoebot* [22, 24, 21]. The name comes from the behavior of the amoebae. Particles are modeled as finite state automata, living and operating in the cells of a hexagonal grid. They are all identical, executing the same distributed algorithm based on local observation of the surroundings, memory and exchanged messages.

Recently, in [21], an approach to homogenize the referred literature on Amoebot has appeared. One of the main intent was to enhance the model with concurrency. Along that line, one of the weakest models for PM, that includes concurrency and eliminates direct communication among particles as well as local and shared memory, is SILBOT [16, 15]. The aim has been to investigate on the minimal settings for PM under which it is possible to accomplish basic global tasks in a distributed fashion. Actually, SILBOT assumes a 2 hops distance visibility for the particles rather than just 1 hop as for Amoebot. It is worth remarking that the observation of the surrounding is the only means for the particles in SILBOT to collect information. Hence such a view must be thought to concern not only other particles in the neighborhood but also any other " visible" information within the sensing range. To that respect, particles in Amoebot are less powerful. However, the information that can be obtained by means of communications (and memory) in Amoebot may concern particles that are very far apart from each other.

In SILBOT, particles operate independently, in a fully asynchronous way, and they admit no persistent memory. Of course, an algorithm designed within such a weak context allows to build a safe, energy-efficient and fault-tolerant system when applied in real scenarios.

In SILBOT, the movement of a particle occurs from one cell to another by alternating between a CONTRACTED state (a particle uniformly occupies a central area of one cell) and an EXPANDED state (a particle occupies one cell but its disposal expresses the intent to move toward a neighboring cell), see, e.g., Figure 1.

Relative positioning among particles is their only (implicit) way of communicating, i.e., stigmergic paradigms are exploited.

Within such a weak system, fundamental tasks have been solved so far, like the *Leader Election* in [15]; the *Scattering* problem in [32], where the particles are required to reach a configuration where they are at distance at least two hops from each other; the *Line formation* in [30, 31]. One of the main open questions posed is about the solvability of other basic primitives.

1.1 Our results

In this paper, we investigate the *Coating* problem in SILBOT. Given a convex object occupying some connected cells of the grid and given a sufficiently large set of particles, the problem asks for a distributed algorithm that moves particles so as to surround the object by occupying all the cells adjacent to it. For the ease of the discussion, we prefer to deal with convex objects. In fact, when dealing with concave objects, still particular cases must be excluded, see, e.g. the so-called *tunnels* as in [26, 20].

We provide a deterministic and distributed algorithm that optimally solves the Coating within $\Theta(n^2)$ rounds, with n being the number of particles, and a round being the minimal time window within which each particle is activated at least once. Moreover, while solving the Coating, our algorithm also resolves the *Hole Compaction*, where a hole is a connected subset of empty cells enclosed by particles and possibly the object, provided that particles are able to recognize holes. An example of hole composed of just one cell can be seen in the top-left of Figure 1, where an empty cell is surrounded by five particles (one of which is expanded) and the object. Furthermore, we prove that the designed algorithm is optimal in terms of number of rounds.

1.2 Related work

The Coating problem has been widely investigated in PM but all the approaches so far are mainly based on stronger assumptions with respect to our algorithm. In particular, the only assumption we share with such works is about chirality, i.e., particles agree on the clockwise and anti-clockwise directions.

The relevance of Coating has been deeply provided for instance in [3] where it is envisioned how PM (intended in terms of swarm of nanobots) in the future might be used to attack cancer cells, surrounding them (i.e., solving the Coating) and then destroy them so as to stop the further growth of the disease. In their approach, first the Leader Election is solved, and then the leader coordinates the other particles so as to accomplish the Coating task for a cave-shaped object.

Another approach used in [4] consists of making the particles able to calculate their distance from the object, and to set an internal value called *Own_Distance*. After each particle have calculated its *Own_Distance*, they send this information to the other particles so that everyone knows each particle's relative distance from the object. Unbounded memory is considered. Then, the particles with the biggest distance from the object send this information to the ones that have a smaller *Own_Distance* so that these ones can make space for the furthest ones and everyone can get closer to the object.

In [20], the Amoebot model is considered. The problem is solved by means of a randomized algorithm, with high probability, within $O(n)$ rounds.

A different model introduced in [26] that reminds PM is the so-called *Pairbot*. In such a work, Coating has been considered and everything revolves around the Pairbot, a single unit formed by two distinguished robots which are each other's *buddy*. It is assumed that initially the pairbots are line-formed and they all agree on the direction where to move in order to reach the object. The movement is guided by the head pairbot which starts by changing state from *short*, where the two buddies are occupying the same cell, to *long*, where the two robots occupy two adjacent cells.

3D Coating has been faced in [35] within the *3D Catom* model [33]. This is a micro-scale lattice-based modular robot that has been investigated also in the context of PM. In fact, modular robots represent a good means for implementing PM theory. They refer to robotic systems where interconnected individual (electro-mechanical) modules can recover from failures or rearrange in order to better adapt to their task-environment (see, e.g. [1, 5, 6, 13, 25, 37, 38]). Recently, the 3D environment has also been approached in [29] for a single agent that has to move tiles in order to cover the surface of an object.

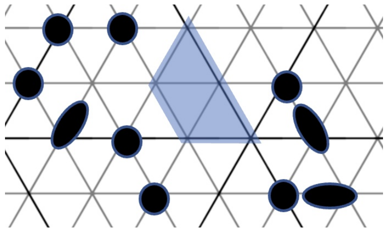

Figure 2 The same configuration of Figure 1 but with the triangular lattice representation. The object is represented by the trapezoidal shaded area; CONTRACTED particles are represented by black circles; EXPANDED particles are represented by black ellipses occupying one node and one neighboring edge.

1.3 Outline

In the next section, we review the SILBOT model in detail and we formalize the Coating problem. In Section 3, we describe and formally present our algorithm for Coating. In Section 4, the correctness proof for the proposed algorithm is presented. Finally, in Section 5, we provide concluding remarks and pose some interesting research directions.

2 SILBOT model and the Coating problem

In this section, we review the SILBOT model for PM introduced in [15], and we formalize the Coating problem.

Operating Environment. Particles operate on an infinite triangular lattice (representing the described hexagonal grid) embedded in the plane, where each node has six incident edges: nodes correspond to hexagonal cells and each edge represents a boundary shared by two cells. Each node can be occupied by at most one particle. There are n particles in the considered system and there is an object occupying a connected and convex set of nodes, see, e.g., Figure 2. From now on we will always refer to the lattice representation, hence cells are referred to as nodes.

Particles and Configurations. Each particle is an automaton with two states, CONTRACTED or EXPANDED (they do not have any other form of persistent memory). In the former state, a particle occupies a single node of the lattice while in the latter, the particle occupies one single node and one of the adjacent edges. Hence, a particle always occupies one node, at any time.

Each particle can sense its surroundings up to a distance of 2 hops, i.e., if a particle occupies a node v, then it can see the neighbors of v and the neighbors of the neighbors of v. Specifically, a particle can determine (i.e., sense) if a node is empty or occupied by a CONTRACTED particle, or occupied by an EXPANDED particle, for each node in its 2-hop visibility range.

Any positioning of CONTRACTED or EXPANDED particles that includes all n particles composing the system plus the object is referred to as a *configuration*.

Furthermore, particles have the so called *exterior awareness*, that informally is the power of detecting what is "outside" the configuration and what is "inside" (that is, *holes*).[2] Formally, a hole is a subset of empty nodes enclosed by particles and possibly by the object.

[2] It is worth remarking that the exterior awareness as well as many other useful information can be obtained in other systems by means of communications combined with memory, both absent in SILBOT.

A particle can distinguish whether a node v within its visibility range is CONT, EXP, IN, OUT or OBJ where: a CONT node is a node occupied by a CONTRACTED particle; an EXP node is a node occupied by an EXPANDED particle; an IN node is an empty node that is part of a hole; an OUT node is an exterior empty node (an empty node that is not part of any hole); an OBJ node is a node occupied by the object. From a practical point of view, the exterior awareness can be implicitly provided by a different level of light among in and out, or a different level of humidity, pressure or the presence/absence of some substance like a liquid or a gas.

Although the exterior awareness might seem a strong assumption, it is worth remarking that the sensing capability of the robots is the only means to acquire information of the surrounding as they cannot exchange messages and have no memory.

Initially, it is assumed that particles are all CONTRACTED and along with the object and possible holes, they constitute a connected set of nodes.

In what follows, we show that starting from any initial configuration, the particles endowed with the simple capabilities described above are able to achieve compaction of holes, if any, while solving Coating. Note that, during the process, the set of particles plus holes plus the object always forms a connected set of nodes.

It is worth remarking that in SILBOT, particles do not have any explicit means of communication. Thus, a particle can acquire information about its surroundings only via its limited view, by means of direct sensing, e.g., weak electromagnetic fields or radars.

Movement and States. As described above, each particle p can occupy only one node v at a time. In order to move to a neighboring node u, p expands on the edge (v, u). Thus, in the EXPANDED state, p occupies node v and edge (v, u) (the physical interpretation is that the particle is occupying one hexagonal cell and has partially entered into the adjacent one, see Figure 1). Note that node u may still be occupied by another particle p'. If p' leaves node u in the future, then p, the EXPANDED particle, will contract into node u during its next activation. There might be arbitrary but finite delays between the actions of these two particles, while the connectivity is still maintained. For example, when p' has moved to another node, edge (v, u) is still occupied by p, the originally EXPANDED particle. In this case, we say that node u is *semi-occupied*. We denote by SO the set of semi-occupied nodes. We ensure that the set of occupied and semi-occupied nodes (plus holes and the object) always induces a connected configuration during the execution of the proposed algorithm.

A very strong constraint of SILBOT, called *Commitment Property*, is that:
A particle commits itself into moving to node u by expanding in that direction, and at the next activation of the same particle, it is constrained to move to node u, if u is empty. A particle cannot revoke its expansion once committed.

Asynchrony and Rounds. The SILBOT model introduces a fine grained notion of asynchrony with possible delays between observations and movements performed by the particles. This reminds the so-called ASYNC model designed for theoretical models dealing with mobile and oblivious robots (see, e.g., [9, 7, 8, 10, 11, 17, 19, 18, 28, 27]). All operations performed by the particles are non-atomic: that is, there can be arbitrarily finite delays between the actions of sensing the surroundings, computing the next decision (e.g., expansion or contraction), executing the decision.

There are no assumptions nor restrictions on the scheduling of these events; thus any possible execution of an actual physical system can be captured by the model. This has important consequences for computability of the particle systems and requires more rigorous techniques for proving correctness of the algorithms (see, e.g. [8, 12, 11]). In particular, algorithms for this model must be inherently simple with a few rules, since this already provides an uncountable large number of possible execution sequences.

A *round* is the time within which all particles have been activated and concluded their activation time at least once. Clearly, the duration of a round is finite but unknown and may vary from time to time.

We include the well-established fairness assumption that each particle must be activated within finite time, infinitely often in any execution of the particle system. Due to the asynchronous nature of the system, it may happen that a particle decides (or is forced, in case of contraction) at time t to take an action, and that this action will actually be executed at time $t' > t$, when other particles might have changed their state; in other words, the action executed at time t' might be based on the obsolete observation of the surrounding taken at time t. The time required to accomplish an action is finite but arbitrary. Hence a round is the shortest time period during which each particle has performed an action (where an action could be the *nil* one if a CONTRACTED particle decides to remain as such, or if an EXPANDED particle finds the target node still occupied).

Orientation and Chirality. We do not make any additional assumptions about the local coordinate systems of a particle but we assume handedness, that is particles agree on *chirality*, i.e., on the clockwise and anti-clockwise directions. Various papers on Coating assume chirality (e.g., [2, 23, 24]). The removal of such an assumption for the Coating problem remains an open problem. Intuitively, in SILBOT, if two sets of particles start surrounding the object along opposite directions, a deadlocked situation might be reached where EXPANDED particles cannot reverse their direction of expansion.

Randomness. We do not assume the possibility of using randomness: particles take deterministic decisions and do not have access to random numbers. Each particle may be activated at any time independently from the others. Once activated, a particle looks at its surrounding (i.e., at its neighbors and at the neighbors of its neighbors) and, on the basis of such an observation, decides (deterministically) its next *action* as follows:

1. A CONTRACTED particle occupying node v, when activated may become EXPANDED, thus occupying node v and one of the edges leading to a neighboring node u, if the edge (v, u) is unoccupied. In this case, we say that the particle *expands* along edge (v, u) and toward node u;
2. An EXPANDED particle occupying node v and edge (v, u), when activated, always *contracts* to u (i.e., moves to u changing its state to CONTRACTED), if u is unoccupied. If u is occupied, then the particle does not change state or location. In other words, as dictated by the Commitment property, a particle in EXPANDED state is obliged to *contract* as soon as it is activated and the destination node is empty.

The activation of each particle is intended to be decided by an "adversarial" scheduler and cannot be controlled by the algorithm designer. If two CONTRACTED particles decide to expand on the same edge simultaneously, exactly one of them (arbitrarily chosen by the scheduler) succeeds. If two particles are EXPANDED along two distinct edges incident to the same node u, and both particles are activated simultaneously, exactly one of them (again, chosen arbitrarily by the scheduler) contracts to node u, while the other particle does not change its EXPANDED state.

Connectivity. An important property we preserve is that the set of particles plus the object plus the set of holes never get disconnected, apart for the aforementioned semi-occupied nodes. As in [15], we allow the particles to move asynchronously while maintaining a " relaxed" sense of connectivity with the object. We remind that when a particle in a node

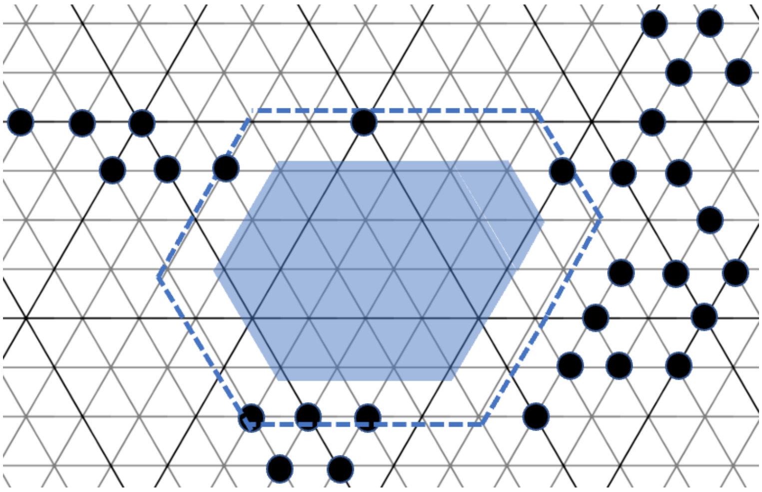

Figure 3 Example of a connected configuration where black circles represent particles; the shaded area includes all the nodes occupied by the object; the dashed line represents EP.

v is EXPANDED along edge (v, u), and u is empty, then u is considered as semi-occupied. Throughout the paper, we say that a configuration is *connected* if the set of nodes that are occupied or semi-occupied or containing the object or holes form a connected subgraph of the lattice. The algorithm we provide to solve the Coating problem always maintains a "relaxed" connected configuration while reduces possible holes and unoccupied nodes surrounding the object.

Given a triangular lattice G, a subgraph of G is *simply connected* if the envelope of its standard planar embedding (i.e. the area delimited by the subgraph) has only one exterior boundary and no interior boundaries (i.e., no holes). A configuration is said to be *simply connected* if the subgraph of lattice G induced by the nodes occupied by particles plus the object is simply connected.

Coating and Holes Compaction Problem. We assume the system is initially in a connected configuration where n particles are all CONTRACTED and sufficient to surround an object obj. Note that obj constitutes any convex subset of connected nodes.

▶ **Definition 1.** *The* External Perimeter *(EP) of obj is considered to be formed by all the nodes adjacent to obj.*

▶ Remark. Due to the above definition and to the assumed chirality, given a node $v \in EP$, the predecessor and the successor of v in EP in the clockwise (and hence also in the anti-clockwise) order are well-defined.

▶ **Definition 2** (Coating and Holes Compaction Problem (CHCP)). *Given an initial configuration of $n \geq |EP|$ particles, an algorithm solves* CHCP *if there exists a time t after which no expansion occurs and the following conditions hold:*

i) *the configuration is simply connected;*

ii) *the external perimeter EP is fully occupied by particles.*

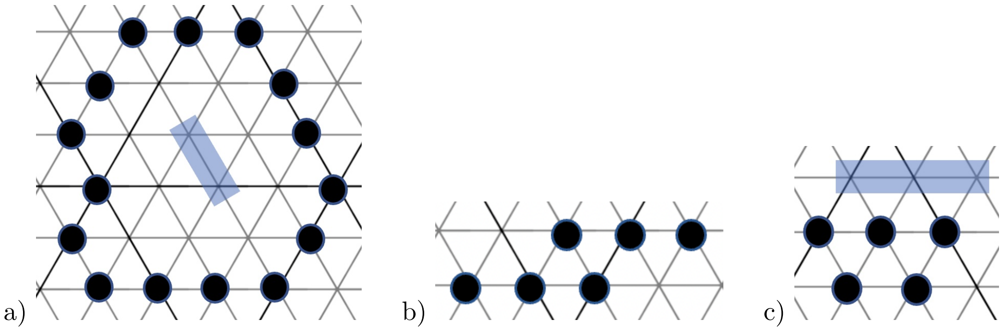

Figure 4 *a*) Possible initial instance where the object (represented by the rectangular shaded area) is completely surrounded by holes, in fact there is no particle in EP; *b*) An instance representing the *bridge problem* where the particles of degree 3 shouldn't expand, especially in opposite directions, as otherwise they may disconnect the particles; *c*) A portion of a possible configuration that requires the rule at Line 12 of Algorithm 1.

3 Algorithm CHC

In this section, we propose an algorithm for CHCP by exploiting the very basic capabilities of the particles dictated by the SILBOT model. In particular, starting from any initial connected configuration of CONTRACTED particles (not necessarily simply connected), the system eventually leads particles to surround *obj* and compact any possible hole.

The pseudocode of Algorithm CHC is reported in Algorithm 1. It is described from the point of view of a single particle. Before analyzing all details, we need the following further notation: given a particle p, we denote by $N(p)$ the set of six nodes adjacent to p, by $N(p, C)$ and $N(p, I)$ the set of CONT and IN nodes, respectively, adjacent to p, and by $N(p, IC)$ the set $N(p, I) \cup N(p, C)$.

The rationale of Algorithm CHC is to make the most external particles EXPANDED toward the internal ones, until " pushing" those adjacent to *obj*. Once particles on EP start to be pushed from the external, they can start moving along EP toward the successive position, according to the ordering provided by Definition 1, e.g., the anti-clockwise direction. In this way, the pushing from the external makes more and more particles move along EP until the Coating problem is solved. In the meanwhile, possible holes are also compacted. A slightly different behavior may occur when, in the initial configuration, *obj* is already surrounded by particles but nodes in EP are not totally occupied, i.e., EP contains some holes, see, e.g., Figure 4.a where EP contains no particles.

Following the aforementioned approach, Algorithm CHC determines two different behaviors according to whether a particle belongs to EP or not:

For particles not in EP, the algorithm allows to expand only particles p such that $G(N(p, IC))$ is connected, there are no semi-occupied nodes in $N(p)$ and $|N(p, IC)| \leq 3$. When a node in $N(p, IC)$ is an empty internal node (i.e., a hole) with $|N(p, I)| = 1$ and p is allowed to expand, then p always expands toward the internal node. In particular, if there exists a CONTRACTED particle p such that $|N(p, IC)| = |N(p, I)| = 1$ (Line 3), then p expands toward the only internal node, while if $|N(p, IC)| = |N(p, C)| = 1$, it expands toward the only CONTRACTED particle.

If $|N(p, IC)| = 2$, instead, then:

i) if there is a node $q \in N(p, I)$, p expands along edge (p, q) (Line 5);
ii) otherwise, it expands toward any CONTRACTED particle (Line 6).

It is worth remarking that holes have priority with respect to CONTRACTED particles.

Algorithm 1 Algorithm CHC (Coating and Holes Compaction).

Require: A connected configuration (including holes and the object) with node p occupied by a CONTRACTED particle.
Ensure: Coating plus Hole Compaction.

1: **if** $p \notin EP$ **then**
2:     **if** $G(N(p, IC))$ is connected $\wedge$ $N(p) \cap SO = \emptyset$ **then**
3:         **if** $N(p, IC) = \{q\}$ **then** Expand along (p, q);
4:         **if** $N(p, IC) = \{q, r\}$ **then**
5:             **if** $q \in N(p, I)$ **then** Expand along (p, q)
6:             **else** Expand along (p, r);
7:         **if** $N(p, IC) = \{r, q, s\}$, with q being between r and s **then**
8:             **if** $\{r, q, s\} \cap N(p, I) = \{x\}$ **then** Expand along (p, x)
9:             **else**
10:                 **if** $|N(r, C)| > 2 \wedge |N(q, C)| > 3 \ \wedge |N(s, C)| > 2$ **then** Expand along (p, q)
11:                 **else**
12:                     **if** $\{r, q\} \subseteq N(p, C) \cap EP \wedge |N(r, C)| = 2 \ \wedge |N(q, C)| > 3 \wedge |N(s, C)| > 2$ **then** Expand along (p, q);
13: **else** let r, p and q be sequential along EP;
14:     **if** $N(p, C) \setminus EP = \emptyset$ **then**
15:         **if** $\exists\ x$ in $N(p) \setminus EP$: x is EXPANDED toward $p \vee r$ in EP is EXPANDED toward p **then** Expand along (p, q);

If $|N(p, IC)| = 3$, then we have three cases:

i) If there is exactly one empty internal node among the three neighbors, then p expands toward it (Line 8);

ii) If the condition at Line 10 is satisfied then p expands toward the central CONTRACTED neighbor (Line 10). This is introduced in order to avoid that in a configuration similar to that depicted in Figure 4.b, particles with degree 3 expand toward opposite directions, hence potentially producing a disconnection. We refer to this situation as the *bridge problem*;

iii) If the condition at Line 12 is satisfied, then p expands toward EP (Line 12). Note that, this case captures a special occurrence as depicted in Figure 4.c. Basically, the presence of *obj* requires a special rule in order to not incur in a deadlock where no particle can expand.

For particles in EP, the algorithm allows the movement of a particle p in EP only if there is at least one particle EXPANDED toward p and all adjacent particles of p not in EP, if any, is EXPANDED. The movement is performed along nodes in EP as well (Line 15). Basically, the idea is that particles in EP tend to occupy new empty nodes of EP only when there is at least one particle which is EXPANDED toward EP and hence eventually moves there. In so doing, a particle in EP remains in EP forever. Note that $|EP|$ particles are required and sufficient to solve the Coating (and Holes Compaction) problem.

It is worth reminding that Algorithm CHC does not deal with EXPANDED particles as those are forced (under the control of the adversarial scheduler) to move toward the chosen direction as soon as possible as dictated by the Commitment property.

In the next section, we show that Algorithm CHC converges to a simply connected configuration where each node in EP is occupied within a finite number of rounds. Note that, the algorithm deals also with (not necessarily simply) connected configurations since there might occur semi-occupied nodes meanwhile. Moreover, the final configuration reached by the algorithm is either composed by all EXPANDED particles and all the nodes in EP occupied, or exactly $n = |EP|$ CONTRACTED particles occupying EP.

4 Correctness

In this section, we show the correctness of Algorithm CHC, i.e., it terminates in a finite number of rounds in a connected configuration where the object is totally surrounded and no holes appear anymore.

We show that, in a connected configuration, there always exists (at least) a particle that can expand, until the Coating (and the Hole Compaction) is solved. Moreover, during the execution of the algorithm, all the generated configurations are guaranteed to be connected (in the "relaxed" way that includes semi-occupied nodes and holes), while the number of empty nodes in EP decreases.

The proof follows an interesting technique where we model each execution of Algorithm CHC as a path in a directed graph $H = (V, E)$, where, given an object, the vertices in V correspond to any connected configuration with $n \geq |EP|$ particles, and the edges in E correspond to transitions among configurations determined by Algorithm CHC. In particular, each vertex $u \in V$ corresponds to a configuration C_v, and there is a directed edge $(v, u) \in E$ if there exists an execution of Algorithm CHC that leads from C_v to C_u, without generating in between further configurations different from C_u.

Note that, each vertex in H only encodes the positions of the particles and not their actual states. I.e., if during an execution of Algorithm CHC a particle p has already decided to expand from a configuration C_v but the expansion occurs after other actions taken by different particles, it means that from C_v there is an edge leading to the configuration C_u where p is EXPANDED and all other particles involved in actions have already computed their actions. However, graph H also contains an edge from C_v to another configuration C_w that differs from C_v only by the expansion of particle p, since this is certainly another possible execution of Algorithm CHC. As we said, graph H represents all possible executions of Algorithm CHC starting from an initial configuration. We do not really need to compute H, it is just intended for analysis purposes. The next theorem exploits the ideal graph H in order to show that Algorithm CHC always terminates and solves the Coating problem.

▶ **Theorem 3.** *Given a convex object obj and $n \geq |EP|$ CONTRACTED particles forming a connected configuration, Algorithm CHC terminates within $\Theta(n^2)$ rounds in a simply connected configuration where EP is totally occupied. Moreover, any configuration generated during the execution of CHC is connected in the relaxed sense.*

Proof. Initially, there are n CONTRACTED particles that along with possible holes and *obj* form a connected configuration. Let $H = (V, E)$ be the directed graph representing the executions of Algorithm CHC as defined above. We prove the correctness of Algorithm CHC by showing the three following properties:

P1: Evolution. Each vertex in H, excluding those corresponding to final configurations, where CHCP is accomplished, has at least one outgoing edge;

P2: Connectivity. H is connected, and hence any configuration obtained by an expansion dictated by CHC is connected;

P3: Acyclicity. Graph H is acyclic.

Note that, the finite number of particles along with the preservation of the connectivity imply that there exists a finite number of configurations. Hence, the three properties guarantee that a final configuration is reached, eventually, where CHCP is accomplished. In fact, P1 guarantees that from any configuration, a different one is generated apart from final configurations. Moreover, by P2 we have that any generated configuration is connected. Since P3 guarantees that no configuration can be generated twice, any execution always leads to a final configuration.

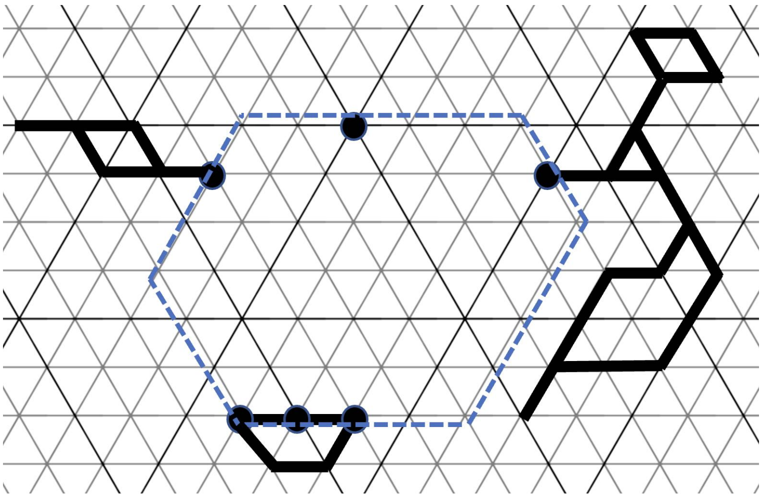

Figure 5 Example of trees of polygons connected via EP, corresponding to the configuration in Figure 3, used in the proof of Theorem 3 for Property P1.

Proof of property P1 (Evolution). If there are semi-occupied nodes, then certainly the configuration will change, eventually. This is due to the fact that EXPANDED particles always complete the committed movement once activated, and due to the fairness of the scheduler. Hence we can assume there are no semi-occupied nodes, i.e., $SO = \emptyset$.

We show that in any (non-final) connected configuration there always exists a CONTRACTED particle p that is allowed to move by algorithm CHC such that:

(i) p is not in EP, $G(N(p, IC))$ is connected and $|N(p, IC)| \leq 3$ or
(ii) p is in EP.

Assume (ii) false and, by contradiction, let us assume each particle p admitting $G(N(p, IC))$ connected, such that $|N(p, IC)| \geq 4$ and hence Algorithm CHC does not allow any expansion. Let us consider the standard planar embedding of the subgraph of lattice G induced by a maximal set of connected CONT and IN nodes, and the envelope containing all nodes of such an embedding. By construction, there might be many of such subgraphs that all together along with obj form a connected configuration, see Figure 5.

Since each considered subgraph is connected, the shape of the corresponding envelope is given by a "tree of polygons", i.e., a set of polygons that are connected by paths of straight lines, possibly of length 0 (i.e., connected via one single point corresponding to a single particle). Each tree is connected ("rooted") to obj – apart for the special cases provided by configurations like in Figure 4.a where obj is contained within a polygon. Moreover, there might be also the case that the mentioned tree of polygons reduces to just one polygon. By hypothesis, the leaves of the tree are not single particles since this would imply that there is at least a particle with only one neighbor. Now, let us consider a leaf of a tree of polygons, and let us assume that it has m vertices (corresponding to m particles). Note that since this polygon is a leaf of the tree, only one of its vertices is connected to the rest of the tree and any other vertex of the polygon corresponds to a particle that has a connected neighborhood.

By hypothesis, we have that any particle has at least 4 neighbors of type CONT or IN. Then, the interior angle of each vertex in the polygon corresponding to these particles measures at least π. Therefore, the sum of the interior angles of the polygon is at least $(m-1)\pi$, which is a contradiction since it is known that such a sum equals $(m-2)\pi$ in any

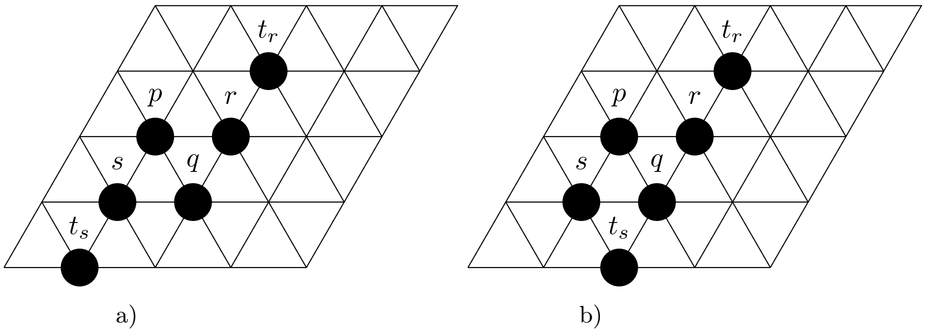

Figure 6 Configurations used in the proof of Theorem 3.

polygon. Hence, we can conclude that there must exist a CONTRACTED particle p such that $G(N(p, IC))$ is connected and $|N(p, IC)| \leq 3$. Furthermore, such a particle belongs to a leaf of a tree of polygons.

Next we show that, given the existence of such a particle, in any non-final configuration where (ii) is false, there exists at least one particle that decides to expand according to Algorithm CHC. In particular, if $|N(p, C)| \leq 2$, then p will expand (see Lines 3-6).

If $|N(p, C)| = 3$, then p is allowed to expand according to three possible conditions dictated by Algorithm CHC. Let r, q and s be the three neighbors of p, with q being the central one. If $|N(r, C)| \leq 2$ ($|N(s, C)| \leq 2$, resp.), then r (s, resp.) will expand, as imposed by Lines 3-6. We remind that we have assumed (ii) false, hence r (s, resp.) is not in EP. By referring to Figures 6.a and 6.b, if $|N(r, C)| > 2$ and $|N(s, C)| > 2$, then r shares neighbor q with p and has a further neighbor, say t_r. Similarly, s is adjacent to p, q, and has a further neighbor t_s. Then two cases can occur: if t_r and t_s are not neighbors of q, then we have a contradiction as the considered polygon is not a leaf of a tree (see Figure 6.a). Otherwise, if at least one among t_r and t_s is a neighbor of q, we have $|N(q, C)| > 3$ and the condition at Line 10 is satisfied (see Figure 6.b), hence p will expand.

About condition at Line 12, it requires that (ii) is true, that is there must exist CONTRACTED particles in EP. If the condition at Line 12 is satisfied, then p expands toward EP (Line 12). Note that this case captures a special occurrence as depicted in Figure 4.c. Basically, the presence of the object requires a special rule in order to not incur in a deadlock. In fact, without such an exception, the particles in the figure would not expand.

If the configuration is non-final but (i) is false and (ii) is true, by the polygon arguments above we have that particles not in EP are all EXPANDED. Moreover, particles cannot be all in EP as otherwise the configuration would be final. We are now going to show that none of the particles at distance 2 from *obj* and adjacent to particles in EP can be expanded by Algorithm CHC toward nodes at distance 3 from *obj*. In so doing, by considering the number of edges among particles at distance 2 from *obj*, we conclude that there must exist at least one particle at distance 2 from *obj* that is EXPANDED toward a particle in EP. Hence, Lines 12-15 are activated and at least one particle in EP expands.

Considering that, for particles not in EP, Algorithm CHC makes particles expanding only toward CONT or IN nodes, let p be a CONTRACTED particle at distance 2 from *obj*, r be a CONTRACTED neighbor of p at distance 3 from *obj* and q be a CONTRACTED neighbor of p in EP. By referring to Figure 7, the first three configurations concern situations where p (the particle at distance 2 from *obj*) cannot expand because $G(N(p, IC))$ is not connected. Whereas, if s is at distance 2 from *obj*, then $G(N(p, IC))$ is connected, hence p would expand toward s. If $|N(p, IC)| > 3$ then p cannot expand. This concludes the proof of Property P1.

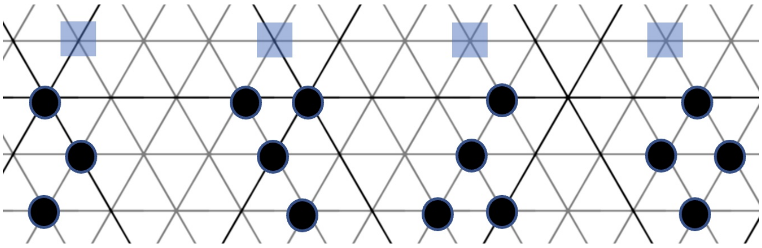

Figure 7 Four possible cases occurring with respect to a particle at distance 2 from the object. In the first three cases, the particle in the middle cannot expand since its neighborhood of CONTRACTED particles is not connected whereas in the last case, Algorithm CHC allows the particles at distance 2 from the object to expand toward each other and, if both activated, by the model only one succeeds.

Proof of property P2 (Connectivity of H). According to Algorithm CHC, we have to analyze when a particle is EXPANDED and whether this (or a combination of such moves) may cause a disconnection.

In fact, the algorithm has been designed to mimic an erosion process from the outer part of the configuration toward obj, plus an oriented movement along EP.

In particular, a particle p not in EP moves toward an IN node v only if p is the central neighbor of the three neighbors of v, say r, p and q, occupied by particles. By the polygon property exploited before, we know that whenever there is a hole, there must exist also a particle in the conditions of p. The movement of p by itself cannot disconnect the configuration. However, this may happen, in principle, because before p actually moves, r and/or q have moved (due to the asynchrony). However, our algorithm guarantees that if r or q move, they can only do it toward the same node v, competing with p.

Movements toward OUT nodes are not allowed, but again a node v may become of type OUT while p has already decided to move toward it. If this is the case, we are guaranteed by the algorithm that the particle that left v won't move again as long as p makes v semi-occupied (see Line 2). Similarly, movements toward nodes occupied by particles cannot cause disconnections.

If $p \in EP$, then p is expanded by the algorithm only if there is another particle expanded toward p, hence the node where p resides may become at most semi-occupied but never purely empty (i.e., without particles expanded toward it), hence again maintaining the configuration connected.

This concludes the proof of Property P2, i.e., H is connected.

Proof of property P3 (Acyclicity of H). Let us associate to each vertex u of H (representing configuration C_u) a pair (α_u, β_u), where α_u is the sum of the number of CONT and IN nodes plus $|SO|$ plus the number of empty nodes in EP in C_u, while β_u is the number of IN nodes plus $|SO|$ plus the number of empty nodes in EP in C_u. We observe that, for each edge (u, v) of H, (α_v, β_v) is lexicographically smaller than (α_u, β_u): in fact, after an action is performed, either at least a CONTRACTED particle in C_u is EXPANDED in C_v (hence the number of CONT nodes decreases) or at least an EXPANDED particle in C_u becomes CONTRACTED in C_v (hence $|SO|$ or the number of IN nodes or the number of empty nodes in EP decreases by at least $k \geq 1$ but the number of CONT nodes increases of the same amount k). Therefore, it is possible to define a topological ordering of the nodes of H as a linear extension of the partial ordering given by the pair (α_i, β_i) of the corresponding configurations C_i.

This concludes the proof of Property P3, i.e., H is acyclic.

Time complexity of Algorithm CHC. About the complexity of the algorithm, each path in H that starts from the node corresponding to the initial configuration and ends in a node corresponding to a final configuration, has a length bounded by the sum of the initial number of CONTRACTED particles plus the number of movements required to occupy EP plus the number of movements necessary to " fill" the initial IN nodes.

The first factor is exactly n. For the second factor, one may think about particles initially disposed in a path shape configuration, touching the object just in one place (say the " head" of the path). Then, the advancement of one step along EP (i.e., toward the final configuration) requires the movement of all the particles in order to not loose connectivity, and consequently the direction toward the object. However, the (adversarial) scheduler may activate all the particles concurrently. In so doing, a whole round is consumed but just one particle expands, namely the " tail" . It follows that when $|EP| = n$ then $O(n^2)$ rounds are required. For the third factor, the number of movements necessary to " fill" the initial IN nodes is upper bounded by $O(n^2)$ with respect to the proposed strategy.

Since an edge of H may correspond to a round of a generic execution, the theorem follows. ◀

Intuitively, the $O(n^2)$ cost of Algorithm CHC in terms of number of rounds turns out to be necessary. In particular, the arguments given for the case of particles disposed in a path shaped configuration provided in the proof of the above theorem turns out to be rather general, i.e., holding for any possible strategy within SILBOT. In fact, when the particles are all aligned and touching the object just in one place, they are forced to proceed one after the other while moving along EP. Any other movement may incur in disconnecting the particles and, or in loosing the direction toward the object, or even worst, in a deadlock with particles expanded in opposite directions. As described above, the sequential movement might be accomplished one step per round. We can then state the following:

▶ **Theorem 4.** *Algorithm* CHC *requires* $\Theta(n^2)$ *rounds for solving the Coating and Holes Compaction problem.*

5 Conclusion

In this paper, we considered the SILBOT model to manage programmable matter. In particular, particles do not have any direct means of communication (i.e., via messages or pebbles), nor local or shared memory, and each particle can retrieve information about its surrounding just up to distance of 2 hops. Moreover, the system is totally asynchronous. This means that a particle can be activated at any time, possibly simultaneously with other particles, and also its actions of expansion and contraction can take different time. All such a freedom is left to the hands of an ideal adversarial scheduler whose only constraint is a kind of fairness, i.e., each particle must be activated within finite time, infinitely often. Finally, particles can only decide on expansions, whereas contractions are automatic and depend on the activations dictated by the scheduler. By adding chirality to the particles we have been able to solve the Coating and Hole Compaction problem. The requirement is to make particles move so as to suitably surround a given static object while obtaining a simply connected configuration. We proposed an optimal algorithm that starting from any initial connected configuration of n particles, eventually leads to solve the problem within $\Theta(n^2)$ rounds.

As future work, one may investigate whether the current assumptions can be further reduced or if other basic tasks can be approached within the same setting. A variant of the Coating problem requires that all the particles contribute to surround the object in a multi-layer fashion. This seems unfeasible within SILBOT with the current setting and gives rise to the investigation on the minimal assumptions for approaching it.

References

1 Hossein Ahmadzadeh, Ellips Masehian, and Masoud Asadpour. Modular robotic systems: Characteristics and applications. *J. Intell. Robotic Syst.*, 81(3-4):317–357, 2016. doi:10.1007/S10846-015-0237-8.

2 Rida A. Bazzi and Joseph L. Briones. Deterministic leader election in self-organizing particle systems. In *Proceedings of the 21st International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 11914. Springer, 2019. doi:10.1007/978-3-030-34992-9_3.

3 Ahmad Reza Cheraghi and Kalman Graffi. A leader based coating algorithm for simple and cave shaped objects with robot swarms. In *5th Asia-Pacific Conference on Intelligent Robot Systems, ACIRS 2020, Singapore, July 17-19, 2020*, pages 43–51. IEEE, 2020. doi:10.1109/ACIRS49895.2020.9162610.

4 Ahmad Reza Cheraghi, Gorden Wunderlich, and Kalman Graffi. General coating of arbitrary objects using robot swarms. In *5th Asia-Pacific Conference on Intelligent Robot Systems, ACIRS 2020, Singapore, July 17-19, 2020*, pages 59–67. IEEE, 2020. doi:10.1109/ACIRS49895.2020.9162617.

5 Anders Lyhne Christensen. *Self-Reconfigurable Robots - An Introduction. Artif. Life*, 18(2):237–240, 2012. doi:10.1162/ARTL_R_00061.

6 Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, and Alfredo Navarra. MOBLOT: molecular oblivious robots. In Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé, editors, *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pages 350–358. ACM, 2021. doi:10.5555/3463952.3463998.

7 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Gathering of robots on meeting-points: feasibility and optimal resolution algorithms. *Distributed Computing*, 31(1):1–50, 2018. doi:10.1007/S00446-017-0293-3.

8 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Asynchronous arbitrary pattern formation: the effects of a rigorous approach. *Distributed Computing*, 32(2):91–132, 2019. doi:10.1007/S00446-018-0325-7.

9 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Embedded pattern formation by asynchronous robots without chirality. *Distributed Computing*, 32(4):291–315, 2019. doi:10.1007/S00446-018-0333-7.

10 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. "Semi-Asynchronous": a new scheduler in distributed computing. *IEEE Access*, 9, 2021. URL: https://ieeexplore.ieee.org/document/9373406, doi:10.1109/ACCESS.2021.3064880.

11 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. Solving the pattern formation by mobile robots with chirality. *IEEE Access*, 9:88177–88204, 2021. doi:10.1109/ACCESS.2021.3089081.

12 Serafino Cicerone, Gabriele Di Stefano, and Alfredo Navarra. A structured methodology for designing distributed algorithms for mobile entities. *Information Sciences*, 574:111–132, 2021. doi:10.1016/J.INS.2021.05.043.

13 Serafino Cicerone, Alessia Di Fonso, Gabriele Di Stefano, and Alfredo Navarra. Molecular oblivious robots: A new model for robots with assembling capabilities. *IEEE Access*, 11:15701–15724, 2023. doi:10.1109/ACCESS.2023.3244844.

14 Sonia Contera. *Nano Comes to Life: How Nanotechnology Is Transforming Medicine and the Future of Biology*. Princeton University Press, 2019.

15 Gianlorenzo D'Angelo, Mattia D'Emidio, Shantanu Das, Alfredo Navarra, and Giuseppe Prencipe. Asynchronous silent programmable matter achieves leader election and compaction. *IEEE Access*, 8:207619–207634, 2020. doi:10.1109/ACCESS.2020.3038174.

16 Gianlorenzo D'Angelo, Mattia D'Emidio, Shantanu Das, Alfredo Navarra, and Giuseppe Prencipe. Leader election and compaction for asynchronous silent programmable matter. In *Proc. 19th Int.'l Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 276–284. International Foundation for Autonomous Agents and Multiagent Systems, 2020. doi:10.5555/3398761.3398798.

17 Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. Gathering on rings under the look-compute-move model. *Distributed Computing*, 27(4):255–285, 2014. doi:10.1007/S00446-014-0212-9.

18 Gianlorenzo D'Angelo, Gabriele Di Stefano, Alfredo Navarra, Nicolas Nisse, and Karol Suchan. Computing on rings by oblivious robots: A unified approach for different tasks. *Algorithmica*, 72(4):1055–1096, 2015. doi:10.1007/S00453-014-9892-6.

19 Gianlorenzo D'Angelo, Alfredo Navarra, and Nicolas Nisse. A unified approach for gathering and exclusive searching on rings under weak assumptions. *Distributed Computing*, 30(1):17–48, 2017. doi:10.1007/S00446-016-0274-Y.

20 Joshua J. Daymude, Zahra Derakhshandeh, Robert Gmyr, Alexandra Porter, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. On the runtime of universal coating for programmable matter. *Natural Computing*, 17(1):81–96, 2018. doi:10.1007/S11047-017-9658-6.

21 Joshua J. Daymude, Andréa W. Richa, and Christian Scheideler. The canonical amoebot model: algorithms and concurrency control. *Distributed Comput.*, 36(2):159–192, 2023. doi:10.1007/S00446-023-00443-3.

22 Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Brief announcement: amoebot - a new model for programmable matter. In *Proc. 26th ACM Symp. on Parallelism in Algorithms and Architectures, (SPAA)*, pages 220–222. ACM, 2014. doi:10.1145/2612669.2612712.

23 Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Universal coating for programmable matter. *Theor. Comput. Sci.*, 671:56–68, 2017. doi:10.1016/J.TCS.2016.02.039.

24 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. In *Proceedings of 21st International Conf. on DNA Computing and Molecular Programming (DNA)*, volume 9211, pages 117–132. Springer, 2015. doi:10.1007/978-3-319-21999-8_8.

25 Toshio Fukuda and Yoshio Kawauchi. Cellular robotic system (CEBOT) as one of the realization of self-organizing intelligent universal manipulator. In *Proc. of the 1990 IEEE Int.'l Conf. on Robotics and Automation, Cincinnati, Ohio, USA, May 13-18, 1990*, pages 662–667. IEEE, 1990. doi:10.1109/ROBOT.1990.126059.

26 Yonghwan Kim, Yoshiaki Katayama, and Koichi Wada. Pairbot: A novel model for autonomous mobile robot systems consisting of paired robots. *CoRR*, abs/2009.14426, 2020. arXiv:2009.14426.

27 David G. Kirkpatrick, Irina Kostitsyna, Alfredo Navarra, Giuseppe Prencipe, and Nicola Santoro. Separating bounded and unbounded asynchrony for autonomous robots: Point convergence with limited visibility. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 9–19. ACM, 2021. doi:10.1145/3465084.3467910.

28 Ralf Klasing, Adrian Kosowski, and Alfredo Navarra. Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. *Theor. Comput. Sci.*, 411:3235–3246, 2010. doi:10.1016/J.TCS.2010.05.020.

29 Irina Kostitsyna, David Liedtke, and Christian Scheideler. Universal Coating in the 3D Hybrid Model, 2023. arXiv:2303.16180.

30 Alfredo Navarra and Francesco Piselli. Brief announcement: Line formation in silent programmable matter. In *Proc. 37th Int.'l Symp. on Distributed Computing (DISC)*, volume 281 of *LIPIcs*, pages 45:1–45:8, 2023. doi:10.4230/LIPICS.DISC.2023.45.

31 Alfredo Navarra and Francesco Piselli. Asynchronous silent programmable matter: Line formation. In *Proc. 25th Int.'l Symp. on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 14310 of *LNCS*, pages 598–612, 2023. doi:10.1007/978-3-031-44274-2_44.

32 Alfredo Navarra, Giuseppe Prencipe, Samuele Bonini, and Mirco Tracolli. Scattering with programmable matter. In *Proceedings of 37th International Conf. on Advanced Information Networking and Applications (AINA)*, Advances in Intelligent Systems and Computing, pages 236–247. Springer, 2023. doi:10.1007/978-3-031-29056-5_22.

33 Benoît Piranda and Julien Bourgeois. Designing a quasi-spherical module for a huge modular robot to create programmable matter. *Auton. Robots*, 42(8):1619–1633, 2018. doi:10.1007/S10514-018-9710-0.

34 Madhuri Sharon, Angelica SL Rodriguez, Chetna Sharon, and Pio Sifuentes Gallardo. *Nanotechnology in the Defense Industry: Advances, Innovation, and Practical Applications*. John Wiley & Sons, 2019.

35 Pierre Thalamy, Benoît Piranda, and Julien Bourgeois. Engineering efficient and massively parallel 3d self-reconfiguration using sandboxing, scaffolding and coating. *Robotics Auton. Syst.*, 146:103875, 2021. doi:10.1016/J.ROBOT.2021.103875.

36 Tommaso Toffoli and Norman Margolus. Programmable matter: Concepts and realization. *Physica D: Nonlinear Phenomena*, 47(1):263–272, 1991. doi:10.1016/0167-2789(91)90296-L.

37 Thadeu Tucci, Benoît Piranda, and Julien Bourgeois. A distributed self-assembly planning algorithm for modular robots. In Elisabeth André, Sven Koenig, Mehdi Dastani, and Gita Sukthankar, editors, *Proc. of the 17th Int.'l Conf. on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 550–558. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018. URL: http://dl.acm.org/citation.cfm?id=3237465.

38 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*, 14:43–52, 2007. doi:10.1109/MRA.2007.339623.