Brief Announcement: Line Formation in Silent Programmable Matter

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

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

Abstract

Programmable Matter (PM) has been widely investigated in recent years. One reference model is certainly Amoebot, with its recent canonical version (DISC 2021). Along this line, with the aim of simplification and to address concurrency, the SILBOT model has been introduced (AAMAS 2020). Within SILBOT, we consider the *Line formation* primitive in which particles are required to end up in a configuration where they are all aligned and connected. We propose a simple and elegant distributed algorithm, optimal in terms of number of movements.

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

Keywords and phrases Programmable Matter, Line formation, Asynchrony, Stigmergy

Digital Object Identifier 10.4230/LIPIcs.DISC.2023.45

Related Version An extended version appears in the Proceedings of the 25th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS), 2023.
Full Version: `https://arxiv.org/abs/2307.16731`

1 Introduction

In the recent years, main attention has been devoted to the so-called *Programmable Matter* (PM). This usually refers to a set of weak and self-organizing computational entities, called *particles*, with the ability to change its physical properties (e.g., shape or color) in a programmable way. Various models have been proposed so far. One that deserves main attention is certainly Amoebot, introduced in [7]. By then, various papers have considered that model, possibly varying some parameters. Moreover, a recent proposal to try to homogenize the referred literature has appeared in [6]. The main intent was to enhance the model with concurrency.

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 [4]. The purpose was to investigate the minimum settings for PM under which basic global tasks can be performed in a distributed manner. Toward this direction, we aim at studying in SILBOT the *Line formation* problem, where particles are required to reach a configuration where they are all aligned (i.e., lie on a same axis) and connected.

The relevance of the Line formation problem is provided by the interest shown in the last decades within various contexts of distributed computing. In graph theory, the problem has been considered in [10] where the requirement was to design a distributed algorithm that, given an arbitrary connected graph G of nodes with unique labels, converts G into a sorted list of nodes. In swarm robotics, the problem has been faced from a practical point of view, see, e.g. [11]. The relevance of line or V-shape formations has been addressed in various

[1] Corresponding author.

37th International Symposium on Distributed Computing (DISC 2023).
Editor: Rotem Oshman; Article No. 45; pp. 45:1–45:8
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

practical scenarios, as in [1, 19]. Most of the work on robots considers direct communications, memory, and some computational power. For application underwater or in the outerspace, instead, direct communications are rather unfeasible and this motivates the investigation on removing such a capability, see, e.g. [12, 17]. Concerning more theoretical models, the aim has been usually to study the minimal settings under which it is possible to realize basic primitives like Line formation. In [2, 16], for instance, Line formation has been investigated for (semi-)synchronized robots (punctiform or not, i.e., entities occupying some space) moving within the Euclidean plane, admitting limited visibility, and sharing the knowledge of one axis of direction. For synchronous robots moving in 3D space, in [18], the plane formation has been considered, which might be considered as the problem corresponding to Line formation for robots moving in 2D. In [13], robots operate within a triangular grid and Line formation is required as a preliminary step for accomplishing the Coating of an object. Within Amoebot, Line formation has been approached in [8], subject to the resolution of the leader election.

2 Definitions and notation

In this section, we review the SILBOT model for PM introduced in [4], and then we formalize the Line formation problem along with other useful definitions.

In SILBOT, particles operate on an infinite triangular grid embedded in the plane. Each node can contain at most one particle. 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 grid while in the latter, the particle occupies one single node and one of the adjacent edges, see, e.g. Figure 1. Hence, a particle always occupies one node, at any time. Each particle can sense its surrounding up to a distance of 2 hops, i.e., if a particle occupies a node v, then it can see the neighbors of v, denoted by $N(v)$, and the neighbors of the neighbors of v. Hence, within its visibility range, a particle can detect empty nodes, CONTRACTED, and EXPANDED particles.

Any positioning of CONTRACTED or EXPANDED particles that includes all n particles composing the system is referred to as a *configuration*. Particles alternate between active and inactive periods decided by an adversarial schedule, independently for each particle.

In order to move, a particle alternates between EXPANDED and CONTRACTED states. In particular, a CONTRACTED particle occupying node v can move to a neighboring node u by expanding along edge (v, u), and then re-contracting on u. Note that, if node u is already occupied by another particle then the EXPANDED one will reach u only if u becomes empty, eventually, in a successive activation. There might be arbitrary delays between the actions of these two particles. When the particle at node u has moved to another node, the edge between v and u is still occupied by the originally EXPANDED particle. In this case, we say that node u is *semi-occupied*.

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.

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 schedule designed for theoretical models dealing with mobile and oblivious robots (see, e.g. [3, 5, 9]). All operations performed by the particles are non-atomic: there can be delays between the actions of sensing the surroundings, computing the next decision (e.g., expansion or contraction), executing the decision.

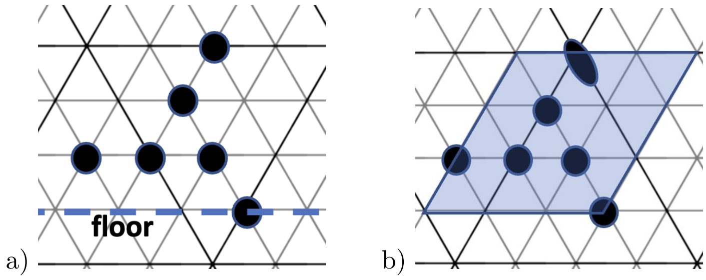

Figure 1 (*a*) A possible initial configuration with emphasized the *floor* (dashed line); (*b*) a possible evolution of the configuration shown in (a) with an expanded particle. The shaded parallelogram is the minimum bounding box containing all the particles.

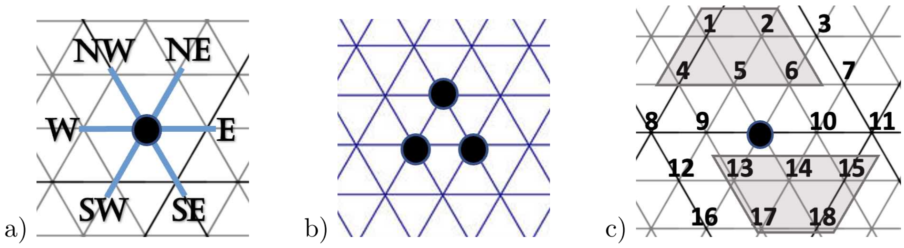

Figure 2 (*a*) A representation of the orientation of a particle; (*b*) An initial configuration where Line formation is unsolvable within SILBOT; (*c*) Enumerated visible neighborhood of a particle; the two trapezoids emphasize two relevant areas for the definition of our algorithm for Line formation.

The well-established fairness assumption is included, where each particle must be activated within finite time, infinitely often, in any execution of the particle system, see, e.g., [9].

Particles are required to take deterministic decisions. Each particle may be activated at any time independently from the others. Once activated, a particle looks at its surrounding (i.e., at 2 hops distance) and, on the basis of such an observation, decides (deterministically) its next *action*.

If two CONTRACTED particles decide to expand on the same edge simultaneously, exactly one of them (arbitrarily chosen by the adversary) succeeds.

If two particles are EXPANDED along two distinct edges incident to a same node w, toward w, and both particles are activated simultaneously, exactly one of the particles (again, chosen arbitrarily by the adversary) contracts to node w, whereas the other particle does not change its EXPANDED state according to the commitment constraint described above.

A relevant property that is usually required in such systems concerns connectivity. A configuration is said to be *connected* if the set of nodes occupied by particles induces a connected subgraph of the grid.

▶ **Definition 1.** *A configuration is said to be* initial*, if all the particles are* CONTRACTED *and connected.*

▶ **Definition 2** (Line formation)**.** *Given an initial configuration, the* Line formation *problem asks for an algorithm that leads to a configuration where all the particles are* CONTRACTED*, connected and aligned.*

▶ **Definition 3.** *Given a configuration C, the corresponding* bounding box *of C is the smallest parallelogram with sides parallel to the West–East and SouthWest–NorthEast directions, enclosing all the particles.*

Table 1 Literature on SILBOT.

Problem	*Schedule*	*View*	*Orientation*	*Reference*
Leader Election	Async	2 hops	no	[4]
Scattering	ED-Async	1 hop	no	[14]
Coating	Async	2 hops	chirality	[15]
Line formation	Async	2 hops	yes	**this paper**

See Figure 1.b for a visualization of the bounding box of a configuration. Note that, in general, since we are dealing with triangular grids, there might be three different bounding boxes according to the choice of two directions out of the three available. As it will be clarified later, for our purposes we just need to define one by choosing the West–East and SouthWest–NorthEast directions. In fact, as we are going to see in the next section, in order to solve Line formation in SILBOT, we need to add some capabilities to the particles. In particular, we add a common orientation to the particles. As shown in Figure 2.a, all particles commonly distinguish among the six directions of the neighborhood that by convention are referred to as the cardinal points $\mathbb{NW}$, $\mathbb{NE}$, $\mathbb{W}$, $\mathbb{E}$, $\mathbb{SW}$, and $\mathbb{SE}$.

Furthermore, in order to describe our algorithm, we need two further definitions that identify where the particles will be aligned.

▶ **Definition 4.** *Given a configuration C, the line of the triangular grid containing the southern side of the bounding box of C is called the* floor*.*

▶ **Definition 5.** *A configuration is said to be* final *if all the particles are* CONTRACTED*, connected and lie on floor.*

By the above definition, a final configuration is also initial. Moreover, if a configuration is final, then Line formation has been solved. Actually, it might be the case that a configuration satisfies the conditions of Def. 2 but still it is not final with respect to Def. 5. This is just due to the design of our algorithm that always leads to solve Line formation on floor.

3 Impossibility results

As shown in the previous section, the SILBOT model is very constrained in terms of particles capabilities. Since its first appearance [4], where the Leader Election problem has been solved, the authors pointed out the need of new assumptions in order to allow the resolution of other basic primitives. In fact, due to the very constrained capabilities of the particles, it was not possible to exploit the election of a leader to solve subsequent tasks. The parameters that can be manipulated have concerned the type of schedule, the hop distance from which particles acquire information, and the orientation of the particles. Table 1 summarizes the primitives so far approached within SILBOT and the corresponding assumptions. Leader Election was the first problem solved when introducing SILBOT [4]. Successively, the Scattering problem has been investigated [14]. It asks for moving the particles in order to reach a configuration where no two particles are neighboring to each other. Scattering has been solved by reducing the visibility range to just 1 hop distance but relaxing on the schedule which is not Async. In fact, the ED-Async schedule has been considered. It stands for Event-Driven Asynchrony, i.e., a particle activates as soon as it admits a neighboring particle, even though all subsequent actions may take different but finite time as in Async. For Coating [15], where particles are required to surround an object that occupies some connected nodes of the grid, the original setting has been considered apart for admitting chirality, i.e., a common handedness among particles.

In this paper, we consider the Line formation problem, where particles are required to reach a configuration where they are all aligned and connected. About the assumptions, we add a common orientation to the particles to the basic SILBOT model. The motivation for endowing the particles with such a capability comes by the following result:

▶ **Theorem 6.** *Line formation is unsolvable within* SILBOT*, even though particles share a common chirality.*

By the assumed orientation, a particle can enumerate its neighborhood, up to distance of 2 hops, as shown in Figure 2.c. This will be useful for the definition of our algorithm.

4 Algorithm *WRain*

The rationale behind the name *WRain* of the proposed algorithm comes by the type of movements allowed. In fact, the evolution of the system on the basis of the algorithm mimics the behavior of particles that fall down like drops of rain subject to a westerly wind. The Line formation is then reached on the lower part of the initial configuration where there is at least a particle – what we have called *floor*.

In order to define Algorithm *WRain*, we need to define some functions, expressing properties related to a node of the grid. We make use of the enumeration shown in Fig. 2.c, and in particular to the neighbors enclosed by the two trapezoids.

▶ **Definition 7.** *Given a node v, the next Boolean functions are defined:*

- $\text{Upper}(v)$ *is* true *if at least one of the visible neighboring nodes from v at positions $\{1, 2, 4, 5, 6\}$ is occupied by a particle;*
- $\text{Lower}(v)$ *is* true *if at least one of the visible neighboring nodes from v at positions $\{13, 14, 15, 17, 18\}$ is occupied by a particle;*
- $\text{Pointed}(v)$ *is* true *if there exists a particle p occupying a node $u \in N(v)$ such that p is* EXPANDED *along edge (u, v);*
- $\text{Near}(v)$ *is* true *if there exists an empty node $u \in N(v)$ such that* $\text{Pointed}(u)$ *is true.*

For the sake of conciseness, sometimes we make use of the above functions by providing a particle p as input in place of the corresponding node v occupied by p.

We are now ready to formalize our Algorithm *WRain*.

Algorithm 1 *WRain.*

Require: Node v occupied by a CONTRACTED particle p.
Ensure: Line formation.

if $\neg Near(v)$ **then**
    if $Pointed(v)$ **then**
        p expands toward $\mathbb{E}$
    else
        if $\neg Upper(v) \land Lower(v)$ **then**
            p expands toward $\mathbb{SE}$

It is worth noting that Algorithm *WRain* allows only two types of expansion, toward $\mathbb{E}$ or $\mathbb{SE}$. Moreover, the movement toward $\mathbb{E}$ can happen only when the node v occupied by a particle is intended to be reached by another particle, i.e., $Pointed(v)$ holds. Another remarkable property is that the algorithm only deals with expansion actions. This is due to the constraint of the SILBOT model that does not permit to intervene on EXPANDED particles, committed to terminate their movement.

5 Correctness and Optimality

In this section, we sketch the proof of correctness of Algorithm *WRain* as well as its optimality in terms of number of moves performed by the particles.

The correctness of Algorithm *WRain* is based on four claims:

Claim 1 - Configuration Uniqueness. Each configuration generated during the execution of the algorithm is unique, i.e., non-repeatable, after movements, on the same nodes nor on different nodes;

Claim 2 - Limited Dimension. The extension of any (generated) configuration is confined within a finite bounding box of sides $O(n)$;

Claim 3 - Evolution guarantee. If the (generated) configuration is connected and not final there always exists at least a particle that can expand or contract;

Claim 4 - Connectivity. If two particles initially neighboring to each other get disconnected, they recover their connection sooner or later (not necessarily becoming neighbors).

The four claims guarantee that a final configuration is achieved, eventually, in finite time, i.e., Line formation is solved. In fact, if from any non-final configuration reached during an execution of *WRain* there is always at least one particle that moves (Claim 3), the subsequent configuration must be different from any already reached configuration (Claim 1). However, since the area where the particles move is limited (Claim 2), then a final configuration must be reached as the number of achievable configurations is finite. Actually, if we imagine a configuration made of disconnected and CONTRACTED particles, all lying on *floor*, then the configuration is not final according to Def. 5 but none of the particles would move. We can prove that such type of configurations cannot occur, and in particular if two particles initially neighboring to each other get disconnected, then they recover their connection, eventually (Claim 4). Since the initial configuration is connected, then we are ensured that also the final configuration is connected as well.

We are now ready to state the correctness and the optimality of *WRain*.

▶ **Theorem 8.** *Given n CONTRACTED particles forming a connected configuration, Algorithm WRain solves Line formation within $\Theta(n^2)$ movements.*

6 Conclusion

We investigated on the Line formation problem within PM on the basis of the SILBOT model. With the aim of considering the smallest set of assumptions, we proved how chirality was not enough for particles to accomplish Line formation. We then endowed particles with a common sense of direction and we proposed *WRain*, an optimal algorithm – in terms of number of movements, for solving Line formation. Actually, it remains open whether by assuming just one common direction is enough for solving the problem. Furthermore, although in the original paper about SILBOT [4], it has been pointed out that 1 hop visibility is not enough for solving the Leader Election, it is worth investigating what happens for Line formation.

Other interesting research directions concern the resolution of other basic primitives, the formation of different shapes or the more general pattern formation problem.

References

1 He Cai, Shuping Guo, and Huanli Gao. A dynamic leader–follower approach for line marching of swarm robots. *Unmanned Systems*, 11(01):67–82, 2023.

2 Jannik Castenow, Thorsten Götte, Till Knollmann, and Friedhelm Meyer auf der Heide. The max-line-formation problem - and new insights for gathering and chain-formation. In *Proc. 23rd Int.'l Symp. on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 13046 of *LNCS*, pages 289–304. Springer, 2021.

3 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.

4 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.

5 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.

6 Joshua J. Daymude, Andréa W. Richa, and Christian Scheideler. The canonical amoebot model: Algorithms and concurrency control. In *Proc. 35th Int.'l Symp. on Distributed Computing (DISC)*, volume 209 of *LIPIcs*, pages 20:1–20:19, 2021.

7 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.

8 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 Andrew Phillips and Peng Yin, editors, *Proc. 21st Int.'l Conf. on Computing and Molecular Programming (DNA)*, volume 9211 of *LNCS*, pages 117–132. Springer, 2015.

9 Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, editors. *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *Lecture Notes in Computer Science*. Springer, 2019. doi:10.1007/978-3-030-11072-7.

10 Dominik Gall, Riko Jacob, Andréa W. Richa, Christian Scheideler, Stefan Schmid, and Hanjo Täubig. A note on the parallel runtime of self-stabilizing graph linearization. *Theory Comput. Syst.*, 55(1):110–135, 2014.

11 Donghwa Jeong and Kiju Lee. Dispersion and line formation in artificial swarm intelligence. In *Proc. 20th Int.'l Conf. on Collective Intelligence)*, 2014.

12 Zhiying Jiang, Xin Wang, and Jian Yang. Distributed line formation control in swarm robots. In *IEEE Int.'l Conf. on Information and Automation (ICIA)*, pages 636–641. IEEE, 2018.

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

14 Alfredo Navarra, Giuseppe Prencipe, Samuele Bonini, and Mirco Tracolli. Scattering with programmable matter. In *Proc. 37th Int.'l Conf. on Advanced Information Networking and Applications (AINA)*, Lecture Notes in Networks and Systems. Springer, 2023.

15 Francesco Piselli. Silent programmable matter: Coating. Master's thesis, University of Perugia, Italy, 2022.

16 Arijit Sil and Sruti Gan Chaudhuri. Formation of straight line by swarm robots. In *Proc. 7th Int.'l Conf. on Advanced Computing, Networking, and Informatics (ICACNI)*, pages 99–111. Springer, 2020.

17 Thomas Sousselier, Johann Dréo, and Marc Sevaux. Line formation algorithm in a swarm of reactive robots constrained by underwater environment. *Expert Syst. Appl.*, 42(12):5117–5127, 2015.

18 Yukiko Yamauchi, Taichi Uehara, Shuji Kijima, and Masafumi Yamashita. Plane formation by synchronous mobile robots in the three-dimensional euclidean space. *J. ACM*, 64(3):16:1–16:43, 2017.

19 Jian Yang, Xin Wang, and Peter Bauer. Line and v-shape formation based distributed processing for robotic swarms. *Sensors*, 18(8):2543, 2018.