The effect of different motion types in simple discrete particle systems with quantitative stigmergy

Gerard Vreeswijk

CS Department, Utrecht University, Utrecht, The Netherlands

ABSTRACT
Discrete particle systems with quantitative stigmergy (ant systems, and particle based simulations of slime mould) are relevant to computational biology and are used as an alternative means to approximate solutions of intractable optimisation problems. The current range of such particle systems exhibits complex behaviour, and particular systems are therefore studied mainly empirically. In contrast, less complex systems, such as cellular automata are better understood and are more amenable to mathematical analysis. To create a bridge between the well-understood area of cellular automata on the one hand and the less understood area of particle systems with quantitative stigmergy on the other hand, this paper proposes to study strongly simplified versions of such particle systems. Eight different motion types are described and evaluated with respect to global system behaviour. The results are analytical as well as empirical. One result is that simple discrete particle systems with quantitative stigmergy permit the derivation of analytical results such as the convergence to a dynamic equilibrium. Another result is that even extreme simplification leaves room for an extraordinary rich spectrum of different motion types, each with its own particular effect on global system behaviour such as particle flow and persistence of particle corridors.
1. Introduction

Recently, much research has been carried out on discrete particle systems with quantitative stigmergy. This includes research on computational models of ant colonies [9,17,18,22,27] and [20,21,47,59,60,67], as well as research on computational models of slime mould [2,36,38,46,58,66,74]. In computer science, discrete particle systems with quantitative stigmergy exhibit a behaviour that is amenable to the modeling of transportation problems and allows one to attack intractable optimisation problems (such as network and traffic problems) in an alternative way [3,39,64].

This paper is based on recent work on particle based computational models of slime mould [2,36,38,46,58,66,74]. Slime mould is a broad term for fungi-like organisms that grow selectively, guided by the locations of food. Currently, most computational models of slime mould are based on the slime mould *Physarum polycephalum* (‘many-headed slime’) [11,44,57]. *Physarum* roughly behaves as follows. If there is no food, it grows exploratively. If there is food, the explorative branches die and the branches toward the food increase their capacity. In the long run, dynamic corridors between food spots are formed in which particles shuttle back and forth to transport and process food. For representative visualisations see for example [3,36–39,66].

There are a number of approaches to model slime mould. Probably the three most important approaches are the flow based approach, the cellular automata based approach, and the particle based approach. In the flow based approach, slime mould is modelled as a number of different streams of which the rate of flow is computed through differential equations [61,73,75,76]. The flow based approach is continuous mathematics and as such fits well within the study of physical systems. Physicists are raised with the idea that gas flows and liquid flows are to be modelled with differential equations. Although physicists do not use continuous mathematics exclusively to model flows, they are at least very much trained to do so. In the cellular automata based approach, slime mould is modelled as a network of cells that are interconnected through states [40,71,72,77]. By nature, the cellular automaton approach is amenable to implementation on parallel hardware, indeed to great success [19]. Finally, in the particle based approach, slime mould is modelled as a constellation of tens of thousands of particles that live on cells, and behave autonomously and interact locally [6,7,31,37,38,62,68]. The particle based approach does not differ much from the cellular automaton based approach. Both approaches belong to the realm of discrete mathematics and virtually all particle based systems can easily be emulated by cellular automata [8,15,70]. Both approaches fit well within disciplines where the simulation of discrete systems is imminent. Such disciplines include biology [29,42], economy [5,23,30] and computer science [33,63].

An appealing property of the particle based approach is that it explores bottom-up, emergent mechanisms of network formation. Recently, particle based models have been used to tackle the travelling salesmen problem to some success [39]. Particle based models also have been used to study the emergence of graph structures that follow the so-called Toussaint hierarchy [1,61]. Indeed, networks formed in particle based models of slime mould tend to converge to so-called Steiner trees. A Steiner tree is a minimum length network to connect a set of vertices, where extra vertices may be added if that helps to reduce the length of the network [3,34,36,37,54]. Particle based models of slime mould have much in common with models of ant colonies or models of termite colonies [citations]. In fact so much so that it is perhaps justified bring all such systems together under the header ‘particle systems with quantitative stigmergy’.

Particle based systems are simple to define, but complex in their behaviour. Because of their complex behaviour, particle based systems are often studied empirically. The objective of such empirical studies is to run experiments after which the emergent behaviour is described and explained qualitatively. As far as I know hardly any analytical results (also known as *theorems*) exist on particle based models of slime mould. Analytical results on ant systems do exist [20,21,47,59,60,67], but these results are to a lesser extent concerned with the formation and persistence of patterns.
1.1. Goal

This paper proposes to study an extremely simplified particle based model of slime mould. This model is based on the model that is discussed in [36], except that almost all real-valued parameters like \((x, y)\) coordinates, heading, sensor offset, sensor angle, rotation angle, and step size are discretised and reduced to trivial values (sensor offset and step size to 1, heading to multiples of \(\pi/4\), sensor angle and rotation angle to \(\pi/4\)). Further, pheromone evaporation is modelled linearly instead of exponentially, and pheromone diffusion is not modelled at all. The consequences of not incorporating diffusion into our model are further discussed in Section 2.3.1. By limiting almost all parameter-values of [36], and by taking diffusion out of the equation, there appears to be a risk that the possible range of patterns that is formed is vastly reduced, so much so that nothing interesting may be left to study. The objective of this paper is to prove otherwise. More in particular my aim is to show that extremely simplified particle based systems form a special category which in itself is worth studying. Extremely simplified particle based systems may then function as some sort of middle ground between simple patch based models such as cellular automata and Langton’s ant on the one hand, and simple agent based models such as multi-turmites systems\(^2\) [13,25,26] and particle swarm systems [14,41,53] on the other hand. With extremely simplified particle systems as our starting point the following four questions are investigated:

1. Can emergent behaviour also be invoked in simplified particle based models? Is it of the same kind as in the original complex models?
2. Do simple models allow the derivation of analytical results?
3. Which motion types in simple models promote macro-properties, such as flow and persistency, flow and persistency?
4. Are macro-properties, such as flow and persistency, influenced by the presence of food?

These questions will be investigated as follows.

1. Runs will be visually inspected, and will be assessed qualitatively on their similarity with runs in comparable but more complex particle based models of slime mould as studied in [31,62,73] and [36,37].
2. Two different motion types with a low level of interactive complexity will be defined. It is my expectation that such simplified models allow the derivation of analytical results (a.k.a. ‘theorems’).
3. Six additional motion types with different collision resolution mechanisms will be defined. All eight motion types will be tested in an experimental setting with respect to flow and persistency. Experiments will be repeated to achieve some degree of reliability.
4. With respect to the presence of food, my ambition is to give descriptive statistics and to draw qualitative conclusions based on that. No statistical inferences will be made. Although making detailed statistical inferences certainly makes sense in the present setting, it currently is beyond the scope of this paper.

The rest of this paper is organised as follows. In Section 2 basic concepts will be explained. In Section 3 the desired emergent behaviour is described. In Section 4, motion types will be discussed that allow co-location of particles. In Section 5, motion types will be discussed that forbid co-location. In Section 6, motion types will be discussed that forbid superimposition (coincidence of location and heading). Section 7–8 discuss experiments and results. Sections 9–10 discuss results, and discuss related work.

2. Basic concepts

This section introduces a simplified particle based model of the slime mould \(P.\) polycephalum. Besides ‘model’, the terms ‘world model’ and ‘model of the world’ will also be used. These terms also help us to remember that certain design choices are made.
2.1. Grid

Particles are supposed to move on a grid. A grid is a finite rectangular array of $m \times n$ patches, $m, n \geq 1$, of which the boundaries are glued together in such a way that the grid can be identified with the surface of a torus. Grid elements are called patches. Between moves, particles reside on the centre of patches.

In the present model, patches store a discrete amount of pheromone $p \in \mathbb{N} \cup \{0\}$. Patches may also store other information to the benefit of the simulating programme, but this information may not be used in the simulation itself.

2.2. Particles

Particles operate autonomously through an infinite sequence of atomic operations. An atomic operation, then, is defined as one complete sense-deliberate-act cycle. Typically, in one atomic operation, a particle senses pheromone, changes heading, moves forward, and drops pheromone. Not necessarily all sub-operations just named are present in one operation cycle, and not necessarily all sub-operations are executed in the order just listed. Furthermore, sub-operations may typically be programmed differently. This admittedly large freedom in the definition of what constitutes to be an ‘operation’ is an issue that will be discussed later (Section 4–6).

Every particle possesses discrete coordinates

$$(x, y) \in \{0, \ldots, m - 1\} \times \{0, \ldots, n - 1\}$$

and a discrete heading

$$h \in \left\{ \frac{k \pi}{4} \mid 0 \leq k < 8 \right\}.$$

So every particle can be oriented straight or diagonally in all eight possible directions. Clearly this is a reduction compared to [36], in which particles can assume all directions. That such a reduction still leaves room for non-trivial behaviour will be demonstrated in the remainder of this paper.

**Definition 2.1:** (Sensoric area, preferred patch, blocking) Let $a$ be a particle.

1. The sensoric area of $a$ consists of the three patches that are indicated in Figure 1.
2. At some point, $a$ selects one of the patches in its sensoric area with maximum pheromone (and resolves ties randomly). The selected patch is called $a$’s preferred patch (at that particular moment).
3. Particle $a$, is blocked by another particle, $b$, if $b$ is on $a$’s preferred patch.
There is a catch to Definition 2.1. If $a$’s sensoric area consists of patches with pheromone 3, 3, and 1, say, then clearly $a$’s preferred patch clearly is either one of the patches with pheromone 3. If another particle, say $b$, is on $a$’s preferred patch, then $a$ is considered to be blocked by $b$, although from our perspective $a$ may just as well opt for the other patch with maximum pheromone 3. These two scenarios are depicted in Figure 2. On the left $a$ (red) has chosen, of the two patches with maximum pheromone, the leftmost patch as its preferred patch (dark green). That patch is occupied by another particle (gray), so $a$ is blocked (hence coloured red). On the right another instantiation of $a$ (yellow) has chosen the rightmost patch of the two patches with maximum pheromone as its preferred patch (dark green). This time the preferred patch is not occupied by another particle, so $a$ is not blocked.

To be sure the notion of blocking is a constraint by design. It is possible to resolve the preferred patch problem in other ways, and doing so would multiply the number of possible move types by (at least) two: one in which the preferred patch is determined before determining the free patches, and one in which the preferred patch (if any) is determined after determining the free patches. In this paper I have chosen to restrict my attention to the last approach.

The sense-deliberate-act cycle of a particle can now be described in more detail as follows.

**Sensoric sub-operation** Generally, particles are supposed to sense pheromone as is illustrated in Figure 1. I say generally, because specific conditions may prevent a particle from sensing in all three directions. These specific restrictions will be discussed later.

One consequence of the current sensoric model is that a particle never senses the amount of pheromone at its own patch. Also particles never stop; they keep moving forward. So, in the current model, particles are heavily ‘forward driven’.

**Motoric sub-operation** After the sensoric stage, particles head towards their preferred patch (Definition 2.1). After that a particle generally is supposed to move to that patch. Again, I say generally, because specific conditions may block the movement of a particle. These specific restrictions will be discussed later.

**Stigmergic sub-operation** After the motoric stage, particles are generally supposed to drop a certain amount of pheromone at their current patch. Again, I say generally, because specific conditions may prevent a particle to drop pheromone. These specific restrictions will be discussed later.

If particles drop pheromone, it is assumed that they drop a fixed quantity $P \in \mathbb{N} \cup \{0\}$, where $P$ is constant throughout the simulation. For the underlying patch with pheromone $p_{old}$ is assumed that
Thus, in the proposed model there is no accumulation of pheromone. Every time a particle hits a patch, the amount of pheromone is simply reset to its maximum, viz. $P$ after which the pheromone evaporates in subsequent time steps. In this way, the underlying grid can ‘remember’ the recency of passages, but not the frequency of passages. Because this pheromone updating scheme still incorporates a notion of evaporation, it can still be classified as a case of *quantitative stigmergy* [69].

### 2.3. Scheduling

Particle operations must be scheduled. I have chosen to do this by means of episodes. An *episode* is a sequence in which all particles operate once. In the current model it is supposed that particles operate asynchronously, *round robin*, which amounts to the following:

- In one episode, every particle operates once, and only once.
- Particles operate in a random order. This random order is determined anew each episode.
- No two particles operate at the same time.

Notice that the present model implies a very strong notion of fairness: no particle operates again until every other particle has operated.

#### 2.3.1. Patch updates

Patch updates can be subdivided into *immediate patch updates* and *episodic patch updates*. The result of a particle dropping pheromone is a typical example of an immediate patch update, since the dropped pheromone is immediately added to the current patch. In general, immediate patch updates happen throughout an episode. Episodic patch updates, on the other hand, take place after a completed episode. Pheromone evaporation is a typical example of an episodic patch update. In our present model, every positive amount of pheromone is decreased by one after each episode. So a linear model of pheromone evaporation is adopted.

Another typical episodic patch update would be pheromone diffusion. However, the notion of diffusion is not incorporated into our model because it would complicate matters considerably from a theoretical point of view. On the other hand, the absence of diffusion also complicates matters from a practical point of view. If there is no diffusion there is no gradient of pheromone against which particles can orient themselves. So it seems that the absence of a diffusion mechanism would heavily impair search. The extent to which this is a problem will become evident from the experiments.

#### 2.3.2. Asynchronicity

Another world model would be to adopt full asynchronicity. In such a world model particles operate asynchronously, with the metaphysical proviso that no two particles operate at the same time. This last assumption is known as the *law of rare events* (usually known from the theory of Poisson distributions [12,35,55]). This alternative world model is more realistic, among other things because it does not imply fairness. However, I have chosen not to adopt this world model because it forces us to calibrate the expected frequency of particle operations with the expected frequency of passive patch updates (like the evaporation of pheromone), the timing of which is quite complicated. So for the sake of simplicity I did not choose to simulate full asynchronicity. I believe this is not a fundamental restriction.

One other way to smuggle true asynchronicity in through the back door (while maintaining a round robin update scheme) can be achieved by letting every particle, when it is its turn, operate (at all) with a fixed probability $0 < p < 1$. Let us call this *circulated asynchronicity*. From the theory of concurrent processes it is well known that processes that operate in circulated asynchronicity in many respects behave similar to processes that operate purely asynchronously [24,49,51,64]. When particles operate
in circulated asynchronicity, it becomes possible that particles operate twice before all other particles have operated, and one obvious consequence would be that the superimposition problem (to be discussed later) resolves naturally.

Still, asynchronicity makes it difficult to invoke controlled behaviour from the particle system as a whole, and makes it much harder to deliver solid statements (theorems) about particle behaviour. In fact, examples may be concocted where particles execute the most exotic patterns in highly unlikely but possible runs. The situation may be compared with the theory of cellular automata in which synchronous cellular automata (like Conway’s game of life [28]), run orderly, and are easier to study than their asynchronous counterparts (cf. [24,49,51]).

Notwithstanding my reservations with respect to the predictability and usefulness of asynchronous models, I have decided to include them in the experiments as motion type 2.

2.3.3. Terminology

The following terminology is used.

1. A run or realisation is a countably infinite sequence of subsequent episodes.

2. Particles $a$ and $b$ are said to co-locate at any particular moment, if the locations of $a$ and $b$ coincide. Co-location is a symmetric relation.

3. A particle $a$ is said to superimpose particle $b$ at any particular moment if the locations and headings of $a$ and $b$ coincide. Superimposition is symmetric as well.

Without too much pre-empting the forthcoming discussion, it can easily be understood, at least intuitively, that superimposition is undesirable. Artificial particles are often programmed uniformly. Therefore particles that superimpose each other will tend do so for long periods of time, especially in environments with little noise and little non-determinism. The problem is that superimposition does not seem to concur well with reality. Particles (or robots) are not supposed to walk on top of each other, at least not for long periods of time. More on this later.

4. A trail of a particle, $a$, is the set of patches of which the amount of pheromone can directly be attributed to (the passage of) $a$.

The trail of a particle $a$ is typically shorter than $P$ because other particles may have crossed patches that were visited by $a$ and contain an amount of pheromone that can be attributed to $a$. If that happens, $a$’s trail is ‘cut off’ to a length strictly less than $P$.

5. An episode interval is any sequence of consecutive positive integers. Episode intervals are typically denoted by $I$.

6. Let $I$ an episode interval and let $a$ and $b$ be particles. Then particle $b$ is said to follow particle $a$ with period $n$ during episode interval $I$ if $b$ walks the trail of $a$, i.e. chooses patches of which the pheromone present can be ascribed to $a$, and the trail distance between $a$ and $b$ is $n$.

It seems that superimposition can be seen as a special case of following, namely, following with zero distance. This is not true. If $b$ follows $a$, then $b$’s behaviour depends on that $a$, while if $b$ is superimposed on $a$, it does not.

7. A run is said to be locked if, eventually, the corresponding particle population consists of followers only.

8. A group of particles is said to be in deadlock if, in the current run, there is certainty that not any one such particle in the group will ever move again. A run is said to have the liveness property if no deadlocks can occur.

3. Goals

It is argued that healthy colonies possess at least two important (but mildly conflicting) properties, viz. flow and persistency [9,17,18,22,27,50]. A high flow roughly means that there is a healthy dynamics in which particles do not get blocked too often but instead are able to move around freely. Further, the idea of flow sometimes includes that particles do not get stuck in one place but instead can be found at
multiple places. Persistency, on the other hand, roughly means that particles on the macro level are not too movable but instead reuse existing pathways, based on the idea that existing pathways lead to resources (‘food’) and generally are tried and true routes to other interesting locations. So persistency basically means a reuse and persistent maintenance (through pheromone) of particle corridors.

Obviously the desire to reuse existing corridors is somewhat at odds with the desire to let particles move freely. So both properties will receive attention.

3.1. Flow

In the present model, flow is already taken care of for a great deal in the way by which sensing and movement are coupled (cf. Figure 1). As a result of this coupling, particles always want to move forward, even if there is maximum pheromone on the patch they reside on. This incorporation of forward-oriented behaviour is very typical in discrete particle based models of biological processes. Such models promote forward driven particle behaviour so as to ensure that no particle by nature wants to camp on specific locations most of the time.

Flow is also promoted by the narrow sensoric area (‘view’) of particles, which decreases the probability that particles ‘hunt their own tail’. Instead, particles have a tendency to bump into the tails of other particles. Of course, a narrow view cannot prevent particles from hunting tails group wise. (i.e. particle 1 hunts particle 2, hunts particle 3, hunts particle 1.) The extent to which this is a problem will become evident from the experiments.

I propose to measure flow through the following variables.

(a) The average number of congestions (due to blocking or stunning) per particle per episode.
(b) The average number of superimpositions per particle per episode.

By no means do these concepts cover all what can be said about flow. They at least give an indication of the dynamics and permeability of particle corridors.

3.2. Persistency

One way to measure persistency is by monitoring the entry of clean patches, i.e. the entry of patches without pheromone. Such an event would indicate that at least one particle in the system does not respect existing corridors but ‘breaks out’ to form a new path (and possibly a new corridor).

**Definition 3.1:** (Perturbation, empirical stability)

(i) A run is said to be perturbed at a particular episode, if, during that episode, at least one particle steps onto a clean patch.
(ii) A run at episode \( n \geq 1 \) is said to be stable for \( k \geq 0 \) episodes if the last perturbation was at episode \( n - k \).

Accordingly, I propose to measure persistency through the following variables.

(a) The average number of perturbations per episode.
(b) The number of episodes that have passed since the last perturbation.

Notice that stability amounts to a dynamic form of stability. i.e. during stable periods particles still move but apparently in such a way that they remain within existing corridors. Also notice that stability amounts to a form of empirical stability, namely, the observation that the system as a whole behaves nicely during a certain number of episodes. Whether empirical stability implies true stability is a difficult analytic question that will not be answered in this paper.

4. Allow co-location

This section studies particle dynamics under the assumption that co-location is allowed. So particles may step on any patch they want. In particular, particles do not have to take into account the location
of other particles. This scenario may be considered as the base case. In fact, it will turn out that the dynamics in the case of co-location is relatively easy to predict.

First, we have the result that, if co-location is allowed, no deadlocks will occur since particles may at all times step on any chosen patch. Next, we have the result that, if co-location is allowed, locking is a kind of ‘dynamic stable state’:

**Theorem 4.1:** When co-location is allowed, every run that is locked will remain locked.

**Proof:** Suppose a run is locked. By definition, all \( N \) particles are followers. I will now argue that any such particle remains a follower. To this end, pick a particle, \( a \), and suppose that \( a \) follows particle \( b \). (It may be that \( a = b \), i.e. that \( a \) hunts its own tail.) There is nothing\(^5\) that makes \( a \) stop following \( b \), except the event that \( a \) happens to cross the trail of another particle, say \( c \), that passed by more recently than \( b \). But then \( a \) starts to follow \( c \) and still is a follower. □

For the next result, we need a lemma. This lemma basically says that independent particles sooner or later become followers.

**Lemma 4.2:** Suppose \( X \) is a finite and path-connected collection of patches with two particles, \( a \) and \( b \), on it. In every run either \( a \) eventually will follow \( b \) or itself.

**Proof:** From the theory of multiple random walks [4,16] it follows that two particles that move in a finite ergodic recurrent class will meet eventually. The current theorem is in effect a consequence of that statement.

With some effort it can be seen that the state space of two particles that move according to Table 1, and that have not met yet, is a finite and transient top-level class, \( T \), that is connected to a finite number of finite classes

\[ \mathcal{A}_1, \ldots, \mathcal{A}_k, \]  

(1)

\( k \geq 1 \), that all represent situations in which \( a \) has become a follower. For example, \( \mathcal{A}_1 \) may represent a situation in which \( a \) follows \( b \) at a distance \( P - 13 \) (and \( b \) is independent), \( \mathcal{A}_2 \) may represent a situation in which \( a \) follows itself at a distance \( P - 27 \) and \( b \) follows \( a \) at a distance \( P - 31 \), etc.

1. Independent particles can delay their walk with one patch at any moment by making a temporary side step at any time. Therefore, two independent particles in any configuration \( T_1 \in T \) can by some ‘external hand’ always be ‘guided’ into any other configuration, \( T_2 \in T \). It follows that \( T \) is a class.
2. Class \( T \) is connected to one of (1). Some of the classes in (1) are transient themselves, because either \( a \) or \( b \) may at some moment cross patches with more pheromone than they are actually following. All further class transitions, however, lead to other classes that remain to express that \( a \) is dependent.

In this way it follows that every process that starts in \( T \) eventually must end in recurrent descendants of (1), which is precisely what is stated in the theorem. □

**Theorem 4.3:** When co-location is allowed, and particles move synchronously, every run will lock eventually.

**Proof:** Suppose that, in a certain run, \( k \) particles are followers, \( 0 \leq k \leq N \), while the remaining \( N - k \) particles are not. By an argument similar to the argument used in Theorem 4.1, the \( k \) followers remain their capacity.

The remaining \( N - k \) particles are not followers. Consider any one such particle. By Lemma 4.2, this particle will hit any one trail eventually (possibly it’s own trail). From then on, this particle is a follower as well, and remains so. □

Thus, if the number of particles is less than the number of patches, and if co-location is allowed, the dynamics eventually converges to particle corridors. Regrettably, further experiments reported in this paper indicate that similar results are much harder to come by in scenarios where co-location is forbidden.
On the other hand, co-location is a problem. When co-location is allowed, and particles operate round robin, particles tend to superimpose (to coincide with respect to location and orientation). This is caused by the fact that the probability that two particles merge on the same trail is many times higher than the probability that two superimposing particles will separate their ways due to non-determinism in the selection of a desired patch. So every pair of particles that superimpose each other tend to remain to do so for extensive amounts of time. Consequently, when co-location is allowed, all runs eventually contain many particle pairs or even particle groups that superimpose each other. Superimposition is further explained and discussed in Section 2.3.3, item 3.

To reduce the number of superimpositions, various alternative particle dynamics will be considered. In Section 5 co-location is forbidden. In Section 6 superimposition is forbidden. To compare the present situation with other motion types, the present motion type (synchronous movement without restrictions on co-location) is incorporated in the experiments as motion type 1.

5. **Forbid co-location**

One way to reduce the number of superimpositions (while sticking to a round robin type of operation scheduling) is to restrict or forbid co-location. This approach gives rise to a range of interesting motion types that all seem to be equally plausible. This section lists a few of them and discusses some of their properties. All the discussed operation types are tested in experiments.

Suppose a particle is blocked by the presence of another particle. Here are some solutions to cope with such a block.

- **Do nothing** Also, do not drop pheromone. This option is not as deadlock prone as one might expect, since blocked particles do not drop pheromone, which in time \( O(P) \) depraves blocked particles from having any patch preferences.

- **Place randomly** Move to a randomly selected empty patch, assume a random heading, and do not drop pheromone. Conceptually this seems to be the most unassuming solution for a block. It is also a conceivable solution. Indeed, spores of many slime mould types (such as *Physarum* and *Dicty*) are spread by wind currents.

- **Move to a sub-optimal patch** Move to a free patch in the sensoric area with the most pheromone. If there is no such patch, move to a randomly selected empty patch (and re-orient). I have no specific intuitions here about the ensuing emergent behaviour.

- **Turn randomly** Assume a random heading, but do not move, and do not drop pheromone. Intuitively, this option seems to provoke noise in the presence of emergent particle corridors (which is undesired).

- **Turn about-face** That is, turn \( \pi \) radials, and do not drop pheromone. This option seems to be an attractive one, because it promises to invoke shuttling on particle corridors. Shuttling is typical behaviour that is observed in real slime mould [36,43,45,65].

- **Iterate move** (Also known as ‘oscillation’ [37] ) Iterate the atomic operation an indefinite number of times, dropping pheromone along the way, until a free patch has been found. This option harbours the risk that particles may get stuck in an infinite loop. To prevent this, a maximum of 100 iterations is allowed. If a particle lands on an occupied patch for the 100th time, it assumes a random orientation with probability 0 < \( p \) ≤ 1. Here, we use \( p = 1/500 \).
Table 1. Motion types.

| Motion type                        | A-synchronous | Live | Co-location | Superimposition | Fall back |
|-----------------------------------|---------------|------|-------------|-----------------|-----------|
| 1. Co-location OK . . . . . . .    | ✓             | ✓    | ✓           | ✓               | -         |
| 2. Circular a-synchronicity . . .  | ✓             | ✓    | ✓           | ✓               | -         |
| 3. Do nothing if blocked . . . . . | .             | ✓    | .           | .               | .         |
| 4. Place randomly if blocked . .  | .             | ✓    | .           | .               | .         |
| 5. Turn to less preferred if blocked | ✓             | ✓    | .           | .               | .         |
| 6. Turn π if blocked . . . . .    | .             | ✓    | .           | .               | .         |
| 7. Iterate move if blocked . . .  | ✓             | .    | .           | .               | .         |
| 8. Do nothing if stunned . . . . . | .             | .    | ✓           | .               | ✓         |

Table 2. Parameter combinations.

| Parameter              | Used values |
|------------------------|-------------|
| World width            | 32, 48      |
| Particle density       | 5, 10       |
| Pheromone dump         | 10, 50, 100 |
| Motion type            | 0, 1, 2, 3, 4, 5, 6, 7 |
| Place food             | false, true |

6. Forbid superimposition

Another way to reduce superimposition would be to redefine the concept of blocking. After all, the objective is not reduce the number of co-locations, but to reduce the number of superimpositions. To this end, the definition of blocking is strengthened.

Definition 6.1: (Stunning) Particle a is stunned by another particle b, if b is on a's preferred patch (this is blocking), and a would superimpose b if a would move to its preferred patch.

Consequently, an interaction between two particles is now considered only problematic if these two particles threaten to superimpose each other in a next move of one of these particles.

Stunning is stronger than blocking. I searched for an adjective that is stronger than 'blocked', hence the terminology. Also, I guess you would be stunned if you would discover that your next move (whatever that may be) would turn you someone else's doppelgänger.

A next motion type now suggests itself: always move to the preferred patch, except when stunned. When stunned, then with probability \( p \) assume a random heading. Else do nothing. Obviously, the magnitude of \( p \) is arbitrary. Currently, we use \( p = 1/500 \), which seems plausible but other values are certainly possible. The motion type that is discussed here is presented as motion type 8 in the experiments.

We now have eight different motion types. (Cf. Table 1) The second column in Table 1, headed ‘Live’, indicates whether runs are guaranteed to be deadlock-free. The fourth column, headed ‘Fall back’, indicates whether the present motion type incorporates a fall back mechanism (like a replacement or a re-orientation or both) if a move is frustrated due to stunning or blocking.

6.1. Behaviour

Appendices 1 and 2 contain impressions (a.k.a. screen shots) of each of the eight different motion types. All runs were executed on a 100 \( \times \) 62 grid with a particle density of 5%. All runs were aborted after 5000 ticks. Red particles are, depending on the move type, either blocked or stunned. The screen shots are supplemented with the number of perturbations and the length of the longest stable period. Both variables are meant to give a rough indication of the volatility of (the process invoked by) each move type. For a definition of the concepts perturbation and stability, cf. Def. 3.1.
7. Experiments

To test the various motion types, I experimented with different starting parameters (cf. Table 2). This gives rise to 192 different parameter combinations. For every parameter combination 5 runs were executed, resulting in a total of 960 runs.

Here follows an explanation of Table 2. Worlds are toroidal surfaces of ratio 16:10, so a world that is 32 patches wide possesses a height of 20. A particle density of 10% on a world of width 32 implies the initialisation of 64 particles. During initialisation, particles are put on separate patches, and if a run involves food, a circular food disk of radius one-twelfth of the world-width is created.

After a run has started, particles move in the manner as described above. Particles dump pheromone with an amount of $P$ every time they manage to step onto a new patch. Otherwise, no pheromone is deposited. After each episode, positive amounts of pheromone on patches are decreased by 1. Prior to each episode, the pheromone of all patches covered by the food spot is reset to $P$.

Runs are terminated after 50,000 stable episodes, or else after 80,000 episodes. After a run ends, the following data is collected.

(a) The average number of congestions (due to blocking or stunning) per particle per episode.
(b) The average number of superimpositions per particle per episode.
(c) The average number of perturbations per episode.
(d) The number of episodes that have passed since the last perturbation.
(e) Whether the run has been terminated due a perceived stability, or due to reaching the maximum number of episodes.

Datum nr. (e) actually depends on datum nr. (d) but is included for convenience.

8. Results

A total of 960 runs were executed. To obtain representative results with respect to motion type and placement of food, cf. Table 2, outcomes of runs are aggregated with respect to world width, particle density and pheromone dump. Again, cf. Table 2. The last three input parameters are considered secondary. They can vary, and they do indeed invoke variation, but it is decided that the variation caused by secondary input variables is to be discounted in the overall assessment of the primary input variables (motion type and placement of food).

Results are summarised as so-called box rankings, cf. the table in Appendix 3. A box ranking is a table where rows represent ‘contenders’ and columns presents ranks (as in a tournament). Each row is a density function of the ranks that a contender scored during a constant number $C$ of tournaments. (Here $C = 2 \times 2 \times 2 \times 3 \times 5 \times 5 \times 5 \times 5 \times 5 \times 5 = 60$.) If table entries are multiplied by $100/C$, then row entries add up to 100 and we speak of a percentage box ranking [32,56]. Motion types are ranked in percentages with respect to the average score of four different dependent variables, viz. the average number of perturbations per episode, by the number of episodes since the last perturbation, the average number of superimpositions per particle per episode, and the average number of congestions per particle per episode. For example, in the first table (average number of perturbations per episode without food), motion type 7 had the lowest average number of perturbations in 66% of all 960 runs. In 25% of all runs motion type 7 had the second lowest number of perturbations. And so on.

The next subsection discusses and explains these box rankings. Due to the emergent nature of the results many of these explanations do not surpass the level of speculation, i.e. many of the explanations given below are mere unverified hypotheses.

8.1. Without food

Motion type 1 and 7 turn out to be relatively the most stable motion types (low average number of perturbations per episode), followed by motion type 2, 6 and 8. I hypothesise that a high average stability is inversely proportional to the frequency of block resolutions, for block resolutions have the
tendency to throw particles off track. For example, motion type 4 is extremely unstable because every random placement of a particle causes noise. On the other hand, motion type 1 is extremely stable and the transient time to stability is short because motion type 1 does not know of blocks at all. As another example, motion type 7 is quite stable because particles are instructed to ‘step over’ blocks and hence are enabled to continue their normal way without disruption in almost all situations.

Motion type 2, 6 and 7 turn out to be relatively the most stable motion types in the long run (high number of episodes since the last perturbation), followed by motion type 1 and 8. I hypothesise that a high ultimate stability is caused by a low frequency of block resolutions combined with a tendency to resolve blocks in identical ways. For example, motion type 6 has a tendency to stabilise in the long run because blocks are resolved deterministically. The same holds of course all the more for motion types 2 and 7 due to the earlier given hypothesis.

Of motion types that allow co-location, motion type 2 generates the lowest average number of superimpositions per particle per episode. I hypothesise that the low number of superimpositions with move type 2 are a result of that fact particles are kept ‘out of sync’, so that any superimposition, would it occur, is volatile. This phenomenon does not occur with the other two motion types that allow co-location, because these other two motion types execute synchronously.

Of motion types that allow superimposition nor co-location, motion type 5 and 6 generate the lowest average number of congestions per particle per episode. I hypothesise that this is caused by the fact that motion type 5 and 6 resolve blocks in ways that are more predictable than with, for example, motion type 4. Motion type 3 also performs badly in this respect because particles involved in a deadlock have the tendency to go out in all directions once all pheromone has evaporated.

8.2. With food

With food, the results are largely similar as to scenarios without food, with the following exceptions. Motion type 1, 2, and 7 turn out to be relatively the most stable motion types in the long run (high number of episodes since the last perturbation), as opposed to motion types 2, 6, and 7 in scenarios without food. The placement of food seems to deteriorate the performance of motion type 6 with regard to long run stability. I hypothesise this is caused by the lack of direction, hence the lack of deterministic behaviour, within food spots. There are more differences. However, these are too detailed to mention. I refer to the overview tables in Section 3.

9. Discussion

The following four questions were investigated:

(1) Whether emergent behaviour in complex models also occurs in simple models.
(2) Whether simple models allow the derivation of analytical results.
(3) Which motion types in simple models promote flow and persistency.
(4) Whether flow and persistency is influenced by the presence of food.

With regard to the first question I think it is fair to state that emergent behaviour in complex models, like the formation of particle corridors and the dynamical adaptation to food, occurs in simple models as well. Visual inspection of the runs support this claim. Of course, the dynamics is not as fine-grained as in the renowned complex models, but the nature of the dynamics appears to be essentially the same.

With regard to the second question my answers are limited. Just prior to Section 8.1 it was already indicated that many explanations are mere hypotheses, and due to the emergent nature of patterning most likely will remain hypotheses. The challenge to transform some of these hypothesis into theorems is interesting but falls beyond the ambition of this paper. As a first step I have proven that particles that move synchronously without restrictions on co-location lock into a stable dynamic equilibrium eventually. Unfortunately, claiming analytical results for other move types proved to be difficult, due to asynchronicity and the complexity induced by block resolution mechanisms.
With regard to the third question I conclude that motion type 7 (iterate if blocked) produces the best flow while leaving existing corridors intact as much as possible. A disadvantage of iteration is that it does not seem to concur well with reality. However, further experimentation reveals that the average number of iterations per particle per episode is modest, namely 2.86 (field width 48, particle density 8, pheromone dump 100). If one still thinks motion type 7 is unrealistic, then, based on the score tables in the appendix, motion type 8 seems to be a good alternative.

With regard to the fourth question, the results point to the conclusion that the presence of food affects flow and persistence adversely. On the other hand, the differences are marginal. To obtain definitive answers, for all 192 different parameter combinations independent t-tests should be performed, with the presence or absence of food as the control variable. When this is done, samples larger than 5 are likely to be necessary to give the test a power that is close to 1. Further, it must be assumed that the dependent variables (the number of perturbations per episode, the number of episodes since the last perturbation, the number of superimpositions per particle per episode, etc.) are normally distributed. The organisation of such statistical experiments makes sense but currently is beyond the scope of this paper.

10. Related work

In [25] it is studied how different motion types affect the evolution of a multi-agent system that consists of multiple particles [48]. In the proposed model all the particles are updated simultaneously and the variation of the updating scheme consists only in choosing different strategies for solving the conflicts produced when two or more particles want to go on the same location. It is shown that for the same formulation of the particles’ behaviour, and the same initial conditions, the use of different updating schemes may lead to qualitatively different evolutions of the system. The work of [25] is relevant to the work presented here because Fates’ work emphasises that slight variations in motion types or updating schemes can exercise a tremendous impact on the overall emergent behaviour of a system.

The study in [13] proposes a discrete dynamic model of multi-agent systems, based on the influence-reaction method of Ferber and Müller [26]. The key idea in [13] is that particles should never act directly on other components of the system (particles or environment) but release influences which are then combined to update the state of the system. Ferber and Müller show that for the same formulation, and the same initial conditions, the use of different simulation schemes may lead to qualitatively different evolutions of the system.

In [10] the robustness of discrete dynamical systems is studied. In particular the behaviour of a lattice-gas models with an alignment-favouring interaction rule is studied. This model exhibits a phase transition between an ordered and a disordered phase. It is shown that the organised phase may result in several different behaviours. It is also shown that behaviours are influenced asymptotically by the definition of the cellular lattice.

The research in [52] deals with the specification and the implementation of multi-level agent based models, using a formal model, IRM4MLS (an Influence Reaction Model for Multi-Level Simulation), based on the influence-reaction principle [26]. Proposed examples illustrate forms of top-down control in (multi-level) multi-agent based-simulations. The work is highly design-oriented and less process oriented. From a thematic perspective, however, it is still relevant to my research.

11. Conclusion

In this paper we have studied eight different motion types in simplified discrete particle based models of the slime mould *P. polycephalum*. We did this to obtain a better understanding of the impact of different motion types on global system behaviour. In particular, the research focused on flow (mobility of particles) and persistency (maintenance of corridors). For an overview of the different motion types see Table 1.
To investigate this question, eight different motion types were defined. Two motion types were studied analytically. The remaining six motion types were studied empirically in different settings.

Based on the analysis and the experiments I conclude that motion type 7 (iterate if blocked) produces the best flow while leaving existing corridors intact as much as possible. If one has objections against the asynchrony nature of motion type 7, then motion type 8 (do nothing if stunned) seems to be an good alternative.

I realise that we have only scratched the surface of the rich interaction models that simplified particle models can provide. More study is needed, both analytically and empirically. Only then can we begin to understand the behaviour of large multi-agent systems.

Software

The software of the simulation and the experiments are contained in one programme and can be downloaded at http://www.projects.science.uu.nl/pipo/software/indexed/ijpeds/2016/01/.

Notes

1. In theory all particle based systems can be emulated by cellular automata, since all cellular automata that matter are Turing complete.
2. The word ‘termite’ is a contraction of the words ‘Turing machine termite’.
3. A set is said to be countably infinite if it can be put into a one-to-one correspondence with the set of positive integers.
4. Earlier, I coined ‘stalk’.
5. Particles possess the same speed, move synchronously, and may co-locate.

Acknowledgements

I would like to thank the three anonymous reviewers for their help. I am especially grateful for pointing out errors and omissions in the exposition and suggesting additional relevant literature.

Disclosure statement

No potential conflict of interest was reported by the author.

References

[1] A. Adamatzky, Developing proximity graphs by Physarum polycephalum: Does the plasmodium follow the toussaint hierarchy? Parallel Process. Lett. 19 (2009), pp. 105–127. Available at http://www.worldscientific.com/doi/abs/10.1142/S0129626409000109.
[2] A. Adamatzky, Physarum Machines: Computers from Slime Mould Vol. 74, Singapore: World Scientific, 2010.
[3] A. Adamatzky and J. Jones, Road planning with slime mould: If physarum built motorways it would route m6/m74 through newcastle, Int. J. Bifurcation Chaos 20 (2010), pp. 3065–3084.
[4] N. Alon, C. Avin, M. Koucký, G. Kozma, Z. Lotker, and M.R. Tuttle, Many random walks are faster than one, Comb. Probab. Comput. 20 (2011), pp. 481–502.
[5] R.M. Axelrod, The Complexity of Cooperation: Agent-based Models of Competition and Collaboration, Princeton (NJ): Princeton University Press, 1997.
[6] M. Becker, Design of fault tolerant networks with agent-based simulation of Physarum polycephalum, 2011 IEEE Congress on Evolutionary Computation, 2011, pp. 285–291.
[7] M. Becker, W. Sarasureeporn, and H. Szczerbicka, Comparison of bio-inspired and graph-theoretic algorithms for design of fault-tolerant networks, ICAS 2012, The Eighth International Conference on Autonomic and Autonomous Systems, Red Hook, NY, 2012, pp. 1–7.
[8] V. Blue and J. Adler, Cellular automata microsimulation of bidirectional pedestrian flows, Transp. Res. Record: J. Transp. Res. Board (1999), pp. 135–141.
[9] E. Bonabeau, Social insect colonies as complex adaptive systems, Ecosystems 1 (1998), pp. 437–443.
[10] O. Bouré, N. Fates, and V. Chevrier, A Robustness Approach to Study Metastable Behaviours in a Lattice-gas Model of Swarming, in: J. Kari, editors. Cellular Automata and Discrete Complex Systems, Berlin: Springer, 2013, pp. 84–97.
[11] E.N. Brewer, Culture and chemical composition of the slime mold physarum polycephalum, Ph.D. thesis, University of Wisconsin, Madison, 1965. Diss. Abstr. 25: 6190–6191.
[12] L.B. Castañeda, V. Arunachalam, and S. Dharmaraja, Introduction to Probability and Stochastic Processes with Applications, New York: Wiley, 2012.
[13] V. Chevrier, N. Fates, *Multi-agent systems as discrete dynamical systems: Influences and reactions as a modelling principle*, Tech. Rep, INRIA Lorraine – LORIA – MAIA, 2008.

[14] M. Clerc, Particle Swarm Optimization Vol 93, New Jersey, US: John Wiley & Sons, 2010.

[15] M. Cook, *Universality in elementary cellular automata*, Complex Syst. 15 (2004), pp. 1–40.

[16] C. Cooper, A. Frieze, and T. Radzik, *Multiple random walks in random regular graphs*, SIAM J. Discrete Math. 23 (2009), pp. 1738–1761.

[17] M. Dorigo, *Ant colony optimization and swarm intelligence*, Proceedings 5th International Workshop, ANTS 2006, September 4–7, 2006, Vol. 4150, Springer, Brussels, Belgium, 2006.

[18] M. Dorigo and L.M. Gambardella, *Ant colony system: A cooperative learning approach to the traveling salesman problem*, IEEE Trans. Evol. Comput. 1 (1997), pp. 53–66.

[19] N. Dourvas, M.A. Tsompanas, G.C. Sirakoulis, and P. Tsalides, *Hardware acceleration of cellular automata Physarum polycephalum model*, Parallel Process. Lett. 25 (2015), p. 1540006–1–1540006-25.

[20] Y. Emek, T. Langner, J. Uttito, and R. Wattenhofer, *Solving the ANTS Problem with Asynchronous Finite State Machines*, Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP), Berlin, 2014.

[21] Y. Emek, T. Langner, J. Uttito, and R. Wattenhofer, *Ants, preprint* (2013). Available at arXiv:1311.3062.

[22] A.P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, New Jersey, US: John Wiley & Sons, 2006.

[23] J.D. Farmer and D. Foley, *The economy needs agent-based modelling*, Nature 460 (2009), pp. 685–686.

[24] N. Fates, *A Guided Tour of Asynchronous Cellular Automata*, in: J. Kari, M. Kutrib and A. Malcher, editors. *Cellular Automata and Discrete Complex Systems*, Berlin: Springer, 2013, pp. 15–30.

[25] N. Fates and V. Chevrier, *How Important are Updating Schemes in Multi-agent Systems? An Illustration on a Multi-turmite Model*, Proceedings of the 9th International Conference on Autonomous Agents and MultiAgent Systems: Volume 1-Volume 1, New York, 2010, pp. 533–540.

[26] J. Ferber and J.P. Müller, *Influences and Reaction: A Model of Situated Multiagent Systems*, Proceedings of Second International Conference on Multi-agent systems (ICMAS-96), Berlin, 1996, pp. 72–79.

[27] J.H. Fewell, *Social insect networks*, Science 301 (2003), pp. 1867–1870.

[28] M. Gardner, *Mathematical games – The fantastic combinations of John Conway's new solitaire game “life”,* Sci. Am. 223 (1970), pp. 120–123.

[29] L. Geris and D. Gomez-Cabrero, *Uncertainty in Biology: A Computational Modeling Approach Vol. 17*, Berlin: Springer, 2015.

[30] G.N. Gilbert, *Agent-based Models, Thousand Oaks (CA): Sage*, 2008.

[31] Y.P. Gunji, T. Shirakawa, T. Niiizato, M. Yamachiyo, and I. Tani, *An adaptive and robust biological network based on the vacant-particle transportation model*, J. Theor. Biol. 272 (2011), pp. 187–200.

[32] Z.C. Holcomb, *Fundamentals of Descriptive Statistics*, Los Angeles, CA: Pyrczak Publishing, 1998.

[33] B. Horling and V. Lesser, *Multiple random walks in random regular graphs*, SIAM J. Discrete Math. 23 (2009), pp. 1738–1761.

[34] M. King and D. Gee, *Multiscale Modeling of Particle Interactions: Applications in Biology and Nanotechnology*, New Jersey, US: John Wiley & Sons, 2010.

[35] M. Kohler and R. Wehner, *Fault-tolerant ants*, in: F. Kuhn, editors. *Distributed Computing*, Berlin: Springer, 2014, pp. 31–45.

[36] C.G. Langton, *Studying artificial life with cellular automata*, Phys. D: Nonlinear Phenom. 22 (1986), pp. 120–149.
[49] J. Lee, F. Peper, and K. Imai, Computing on a simple asynchronous cellular automaton, 2013 First International Symposium on Computing and Networking (CANDAR), 2013, pp. 505–509.

[50] P. Luykx, Termite colony dynamics as revealed by the sex-and caste-ratios of whole colonies of incisitermes schwarzi banks (isoptera: Kalotermitidae), Insectes Soc. 33 (1986), pp. 221–248.

[51] L. Manzoni, Asynchronous cellular automata and dynamical properties, Natural Comput. 11 (2012), pp. 269–276.

[52] G. Morvan and D. Jolly, Multi-level agent-based modeling with the influence reaction principle, preprint (2012). Available at arXiv:1204.0634.

[53] R. Poli, J. Kennedy, and T. Blackwell, Particle swarm optimization, Swarm Intell. 1 (2007), pp. 33–57.

[54] H.J. Prömel and A. Steger, The Steiner Tree Problem: A Tour through Graphs, Algorithms, and Complexity, Berlin: Springer Science & Business Media, 2012.

[55] S.M. Ross, Stochastic Processes, New York: Wiley, 1996.

[56] R.P. Runyon, Descriptive and Inferential Statistics: A Contemporary Approach, Menlo Park, CA: Addison-Wesley, 1977.

[57] H.W. Sauer, Developmental Biology of Physarum, London: Cambridge University Press, 1982.

[58] A. Schumann, K. Pancerz, and J. Jones, Towards logic circuits based on Physarum polycephalum machines – The ladder diagram approach, BIOVICES 2014 – Proceedings of the International Conference on Biomedical Electronics and Devices, ESEO, 3–6 March, 2014, SciTePress, Angers, Loire Valley, France, 2014, pp. 165–170. doi:10.5220/0004839301650170.

[59] A. Shiloni, N. Agmon, and G.A. Kaminka, Of robot ants and elephants, Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, New York, 2009, pp. 81–88.

[60] A. Shiloni, A. Levy, A. Felner, and M. Kalech, Ants meeting algorithms, Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 2-Volume 2, New York, 2010, pp. 567–574.

[61] T. Shirakawa, A. Adamatzky, Y.P. Gunji, and Y. Miyake, On simultaneous construction of voronoi diagram and delaunay triangulation by Physarum polycephalum, Int. J. Bifurcation Chaos 19 (2009), pp. 3109–3117.

[62] T. Shirakawa and Y.P. Gunji, Computation of voronoi diagram and collision-free path using the plasmodium of Physarum polycephalum, Int. J. Unconventional Comput. 6 (2010), pp. 79–88.

[63] Y. Shoham and K. Leyton-Brown, Multiagent Systems: Algorithmic, Game-theoretic, and Logical Foundations, London: Cambridge University Press, 2008.

[64] S. Singh Dhillon, Ant routing, searching and topology estimation algorithms for ad hoc networks, Ph.D. thesis, Technische Universiteit Delft, IOS Press, 2008.

[65] S. Sommer and R. Wehner, Vector navigation in desert ants, cataglyphis fortis: Celestial compass cues are essential for the proper use of distance information, Naturwissenschaften 92 (2005), pp. 468–471.

[66] E. Strano, A. Adamatzky, and J. Jones, Physarum itinerae: Evolution of roman roads with slime mould, Int. J. Nanotechnol. Mol. Comput. 3 (2011), pp. 31–55. doi:10.4018/jnmc.2011040103.

[67] N.R. Sturtevant, V. Bulitko, and Y. Björnsson, On learning in agent-centered search, Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1-Volume 1, New York, 2010, pp. 333–340.

[68] A. Takamatsu, E. Takaba, and G. Takizawa, Environment-dependent morphology in plasmodium of true slime mold Physarum polycephalum and a network growth model, J. Theor. Biol. 256 (2009), pp. 29–44.

[69] G. Therauzal and E. Bonabeau, A brief history of stigmergy, Artif. Life 5 (1999), pp. 97–116.

[70] T. Toffoli, Cellular automata machines as physics simulators, International conference on the impact of digital microelectronics and microprocessors on particle physics, Singapore, 1988.

[71] M.A.I. Tsompanas and G.C. Sirakoulis, Modeling and hardware implementation of an amoeba-like cellular automaton, Bioinspiration Biomimetics 7 (2012), p. 036013.

[72] M.A.I. Tsompanas, G.C. Sirakoulis, and A.I. Adamatzky, Evolving transport networks with cellular automata models inspired by slime mould, IEEE Trans. Cyber. 45 (2015), pp. 1887–1899.

[73] S. Tsuda, M. Aono, and Y.P. Gunji, Robust and emergent physarum logical-computing, Biosystems 73 (2004), pp. 45–55.

[74] S. Tsuda and J. Jones, The emergence of synchronization behavior in physarum polycephalum and its particle approximation, Biosystems. 103 (2011), pp. 331–341. doi:10.1016/j.biosystems.2010.11.001.

[75] T. Umedachi, R. Idei, K. Ito, and A. Ishiguro, A fluid-filled soft robot that exhibits spontaneous switching among versatile spatiotemporal oscillatory patterns inspired by the true slime mold, Artif. Life 19 (2013), pp. 67–78. doi:10.1162/ARTL_a_00081.

[76] T. Umedachi, K. Takeda, T. Nakagaki, R. Kobayashi, and A. Ishiguro, Fully decentralized control of a soft-bodied robot inspired by true slime mold, Biol. Cybern. 102 (2010), pp. 261–269.

[77] X. Zhang, A. Adamatzky, F.T. Chan, Y. Deng, H. Yang, X.S. Yang, M.A.I. Tsompanas, G.C. Sirakoulis, and S. Mahadevan, A biologically inspired network design model, Sci. Rep. 5 (2015), pp. 1–15.
Appendix 1. Visualisation of motion types without food

Figure A1. Move type 1: 105 perturbations, length of longest stable period is 4715.

Figure A2. Move type 2: 192 perturbations, length of longest stable period is 2925.

Figure A3. Move type 3: 4775 perturbations, length of longest stable period is 32.

Figure A4. Move type 4: 4974 perturbations, length of longest stable period is 1.

Figure A5. Move type 5: 4998 perturbations, length of longest stable period is 1.

Figure A6. Move type 6: 165 perturbations, length of longest stable period is 2874.

Figure A7. Move type 7: 83 perturbations, length of longest stable period is 4868.

Figure A8. Move type 8: 1460 perturbations, length of longest stable period is 2740.
Appendix 2. Visualisation of motion types with food

Figure B1. Move type 1: 1199 perturbations, length of longest stable period is 142.

Figure B2. Move type 2: 2571 perturbations, length of longest stable period is 77.

Figure B3. Move type 3: 4734 perturbations, length of longest stable period is 16.

Figure B4. Move type 4: 4971 perturbations, length of longest stable period is 1.

Figure B5. Move type 5: 4998 perturbations, length of longest stable period is 1.

Figure B6. Move type 6: 1197 perturbations, length of longest stable period is 174.

Figure B7. Move type 7: 1032 perturbations, length of longest stable period is 236.

Figure B8. Move type 8: 2837 perturbations, length of longest stable period is 50.
### Appendix 3. Percentage box ranking of motion types

#### Without food

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 8   | 66  | 25  |     |     |     |     |     |
| MT 2 | 25  | 58  | 8   |     |     |     |     |     |
| MT 3 | 8   | 16  | 66  | 8   |     |     |     |     |
| MT 4 | 8   | 75  | 16  |     |     |     |     |     |
| MT 5 | 83  | 8   |     |     |     |     |     |     |
| MT 6 | 8   | 25  | 41  | 25  |     |     |     |     |
| MT 7 | 66  | 25  |     |     |     |     |     |     |
| MT 8 |     |     |     |     |     |     |     |     |

#### Ordered by the average number of perturbations per episode:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 8   | 41  | 16  | 8   | 25  |     |     |     |
| MT 2 | 33  | 25  | 33  | 8   |     |     |     |     |
| MT 3 | 8   | 83  | 8   |     |     |     |     |     |
| MT 4 | 8   | 91  |     |     |     |     |     |     |
| MT 5 |     |     |     |     | 8   | 91  |     |     |
| MT 6 | 25  |     | 16  | 50  | 8   |     |     |     |
| MT 7 | 25  | 25  | 41  | 8   |     |     |     |     |
| MT 8 | 8   | 8   | 8   | 58  | 16  |     |     |     |

#### Ordered by the number of episodes since the last perturbation:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 8   | 41  | 16  | 8   | 25  |     |     |     |
| MT 2 | 33  | 25  | 33  | 8   |     |     |     |     |
| MT 3 | 8   | 83  | 8   |     |     |     |     |     |
| MT 4 | 8   | 91  |     |     |     |     |     |     |
| MT 5 |     |     |     |     | 8   | 91  |     |     |
| MT 6 | 25  |     | 16  | 50  | 8   |     |     |     |
| MT 7 | 25  | 25  | 41  | 8   |     |     |     |     |
| MT 8 | 8   | 8   | 8   | 58  | 16  |     |     |     |

#### Ordered by the average number of superimpositions per particle per episode:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 91  | 8   |     |     |     |     |     |     |
| MT 2 | 8   | 91  |     |     |     |     |     |     |

#### With food

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 8   | 66  | 25  |     |     |     |     |     |
| MT 2 | 25  | 58  | 8   |     |     |     |     |     |
| MT 3 | 33  | 58  | 8   |     |     |     |     |     |
| MT 4 | 16  | 50  | 33  |     |     |     |     |     |
| MT 5 | 83  | 16  |     |     |     |     |     |     |
| MT 6 | 8   | 25  | 50  | 8   |     |     |     |     |
| MT 7 | 41  | 58  |     |     |     |     |     |     |
| MT 8 | 75  | 16  |     |     |     |     |     |     |

#### Ordered by the average number of perturbations per episode:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 8   | 8   | 41  | 16  | 8   | 25  |     |     |
| MT 2 | 33  | 25  | 33  | 8   |     |     |     |     |
| MT 3 | 8   | 83  | 8   |     |     |     |     |     |
| MT 4 | 8   | 91  |     |     |     |     |     |     |
| MT 5 |     |     | 8   | 91  |     |     |     |     |
| MT 6 | 25  | 16  | 50  | 8   |     |     |     |     |
| MT 7 | 25  | 25  | 41  | 8   |     |     |     |     |
| MT 8 | 8   | 8   | 8   | 58  | 16  |     |     |     |

#### Ordered by the number of episodes since the last perturbation:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 50  | 25  | 16  | 8   |     |     |     |     |
| MT 2 | 25  | 33  | 25  | 16  |     |     |     |     |
| MT 3 | 8   | 16  | 50  | 8   |     |     |     |     |
| MT 4 | 25  | 66  | 8   |     |     |     |     |     |
| MT 5 | 25  | 75  |     |     |     |     |     |     |
| MT 6 | 8   | 50  | 41  |     |     |     |     |     |
| MT 7 | 25  | 33  | 33  | 8   |     |     |     |     |
| MT 8 | 16  | 16  | 41  | 25  |     |     |     |     |

#### Ordered by the average number of superimpositions per particle per episode:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 1 | 83  | 16  |     |     |     |     |     |     |
| MT 2 | 16  | 83  |     |     |     |     |     |     |

#### Ordered by the average number of congestions per particle per episode:

| Rank | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| MT 3 | 100 |     |     |     |     |     |     |     |
| MT 4 | 100 |     |     |     |     |     |     |     |
| MT 5 | 33  | 66  |     |     |     |     |     |     |
| MT 6 | 100 |     |     |     |     |     |     |     |
| MT 7 | 66  | 33  |     |     |     |     |     |     |
| MT 8 | 100 |     |     |     |     |     |     |     |