Robustness of Multi-agent Models: The Example of Collaboration between Turmites with Synchronous and Asynchronous Updating

Selma Belgacem

LITIS Laboratory
Rouen University, France
selma.belgacem@etu.univ-rouen.fr

Nazim Fatès

Inria Nancy – Grand Est & LORIA
615 rue du Jardin botanique
54 600 Villers-lès-Nancy, France
Corresponding author: nazim.fates@loria.fr

The robustness of multi-agent systems to simulation conditions is analyzed through a precise example, invented by Langton to investigate the foundations of artificial life. This system is composed of simple and memoryless agents, the turmites, which obey simple discrete local rules. While the local rules that govern each agent are kept constant, the interaction between agents is modified through nine variations. Our method consists in varying the updating scheme (synchronous vs. asynchronous) and the local conflict resolution policy (strong or weak exclusion rules). The effect of these modifications on three collaborative phenomena is experimentally estimated. The macroscopic robustness of the system is analyzed by examining how the conflicts that occur at the microscopic scale generate diverging trajectories of the system. Observations confirm that the definition of the individual agent's behavior is not the only setting that matters in the emergence of collaborative phenomena in complex systems: the way the agents are updated is also a key choice.

1. Introduction

In a pioneering paper on artificial life published in 1986, Langton stated [1]: "A common aggregate organization in nature is that of *society*. The global behavior of a society is an emergent phenomenon, arising out of all of the local interactions of its members. [...] We know that complex behavior can emerge from the interaction of very simple parts. Colonies of social insects provide a good subject material to the study of artificial life because they so readily exhibit complex behavior emerging from the interaction of very simple living

https://doi.org/10.25088/ComplexSystems.21.3.165

parts." The method followed by Langton was very close to Turing's work on morphogenesis [2]: Instead of trying to capture life's complexity by building more and more realistic models, which is a never-ending task, simplifying the model as much as possible can help us identify the mechanisms that are sufficient for a phenomenon to appear.

Langton proposed considering a simple system composed of one or several agents that operate on a two-dimensional grid. (Bunimovich and Troubetzkoy studied an equivalent system in the context of particle systems [3].) Each cell of the grid has one state: 0 or 1. The system is updated in discrete time and the agents, now known as *Langton's ants* or *turmites* [4], are memoryless and follow two simple symmetric laws: (*a*) If the turmite is on a cell in state 0, the cell state flips to 1; the turmite turns left and advances to the next cell. (*b*) If the turmite is on a cell in state 1, the cell state flips to 0; the turmite turns right and advances to the next cell.

Langton observed that the behavior of a single turmite was already a puzzling phenomenon. In the case where several agents were put together, interesting collaborative phenomena could emerge and lead to the construction of drastically different patterns than those observed for a single agent. This was interpreted as an emergence of collaboration between agents, a topic that is now widely considered a key problem in many sciences. In biology, for instance, it is still a challenge to understand how social insects may collaborate to construct their nests [5]. However, as Langton himself admits: "There are so many ways that these virtual ants can encounter one another that the transition rules have not yet been worked out for all of the possible encounters." Our goal in this paper is to complete the work of Langton and his followers by broadening the way interactions between ants are defined.

We aim to discover not only novel collective phenomena, but also wish to gain insight into how much the global behavior of the system depends on these local interactions.

1.1 Turmites

Dynamics of turmites have been well studied when only a single turmite or particle is considered [6–8].

Starting from an initial grid with all cells in state 0, the turmite follows an irregular trajectory for approximately 10 000 steps and then suddenly enters into a periodic behavior. This behavior leads to the formation of a regular translating structure called a *path* (also known as a "highway"; see Figure 1). Different generalizations [9, 10] have also been proposed, but, surprisingly enough, systems with multiple turmites have been much less explored so far. The only results we are aware of are the studies by Chopard and Droz [11] and by Beuret and Tomassini [12].

https://doi.org/10.25088/ComplexSystems.21.3.165

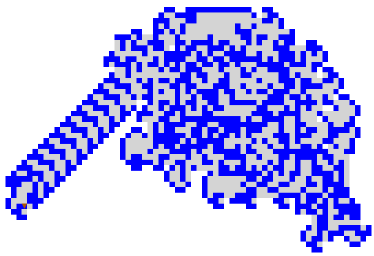


Figure 1. Evolution of a system with a single turmite starting from an empty environment: the turmite draws an infinite path. White cells are unvisited 0-cells, gray cells are visited 0-cells, and the blue/dark cells are 1-cells. This convention is kept in the remainder of the paper.

1.2 Nine Variations on One Rule

One possible reason the multi-turmite system has been scarcely studied is that introducing multiple turmites also produces ambiguities. Indeed, it is not clear from the local rule how to decide in which order (if any) to update agents and how to solve their potential conflicts when they share the same target cell. In some cases, these ambiguities may even render experiments difficult to reproduce.

To tackle these difficulties, a method was proposed by Chevrier and Fatès as a specification of Ferber and Muller's influence–reaction paradigm [13]. It consists of describing multi-agent systems as discrete dynamical systems [14]. Each description is obtained with a *simulation scheme*, that is, a particular way of updating components and a particular method for solving the potential conflicts that would appear during this updating. As a result, even when using the same model and when starting from the same initial condition (the theme), the use of different simulation schemes (the variations) may produce several qualitative behaviors.

In the case of cellular automata, after the pioneering observations by Ingerson and Buvel [15], a number of studies have shown that an asynchronous update leads to the observation of a wide range of surprising phenomena (see [16–18] for recent references). Our purpose is to present a similar study for a simple multi-agent system. We consider different ways of dealing with the spatial conflicts that appear when multiple agents need to share the same location and examine the differences produced by a synchronous or an asynchronous update.

1.3 From Artificial Life to Natural Phenomena

Although the system proposed by Langton is simplistic, it may help us evaluate the effects of implicit choices in the simulation of more complex systems. For instance, if a model of biological systems such as viruses or bacteria is needed, it may become possible to estimate sepa-

https://doi.org/10.25088/ComplexSystems.21.3.165

rately how much of the observed behavior is due to the internal dynamics of cells and how much is due to the interactions between cells. Moreover, in discrete systems, it is often difficult to decide whether to use a synchronous model (e.g., lattice-gas cellular automata) or an asynchronous model (e.g., interacting particle systems). In this paper, we do not decide a priori but rather propose testing different simulation scenarios and comparing them from a phenomenological point of view.

The outline of the paper is as follows. Section 2 is devoted to the definition of the multi-turmite system and the different simulation schemes studied. In Section 3, three emergent phenomena are presented and we study their robustness to asynchrony with a macroscopic approach. A microscopic analysis is carried out in Section 4 to understand this robustness in more detail. Finally, the questions opened by these observations are discussed in Section 5.

2. Foundations

Before examining the outcome of simulations, let us first formally define our multi-turmite system. As shown in the following, the operation is not as straightforward as it may first seem. Our presentation of the dynamical system follows the method of Chevrier and Fatès [14], although we skip here all the intermediary steps of the method for the sake of conciseness.

We denote by $\mathcal{L} = \mathbb{Z}^2$ the grid (or lattice). Each cell $c \in \mathcal{L}$ has a state in $Q = \{0, 1\}$. The overall grid state is denoted by $\mathbf{S} \in Q^{\mathcal{L}}$. Let N be the number of turmites; we denote by $\mathrm{T} = \{1, \ldots, N\}$ the set of turmites. Each turmite i has a position $P_i \in \mathcal{L}$ and an orientation $O_i \in \mathcal{D} = \{\mathtt{N}, \mathtt{E}, \mathtt{S}, \mathtt{W}\}$ associated to the directions north, east, south, and west.

We denote by $\mathbf{P} = (P_1, \ldots, P_N) \in \mathcal{L}^N$ and by $\mathbf{O} = (O_1, \ldots, O_N) \in \mathcal{D}^N$ the N-tuple of turmite positions and orientations, respectively. The state of a system is a *configuration* that is represented by a triplet $\sigma = (\mathbf{S}, \mathbf{P}, \mathbf{O}) \in \Sigma = Q^{\mathcal{L}} \times \mathcal{L}^N \times \mathcal{D}^N$. Using these notations, we describe our multi-turmite system as a discrete dynamical system on Σ, that is, advancing by one time step corresponds to applying Γ, the *global transition function*: $\Gamma : \Sigma \to \Sigma$.

Now, consider the problem of formally describing Γ based on the informal description of turmite behavior stated earlier and given that:

- we want to update the turmites according to three temporal schemes; and
- we want to examine various ways of solving the conflicts that appear when multiple turmites simultaneously want to move onto the same cell.

Our proposition consists of defining Γ with two auxiliary functions. The first is the *updating method* Δ and the second is the *conflict reso-*

https://doi.org/10.25088/ComplexSystems.21.3.165

lution policy ξ. We denote by $\Gamma_{\Delta,\xi}$ the global updating function obtained with an updating method Δ and a conflict resolution policy ξ. The system's *orbit* (or trajectory) is the sequence of configurations $(\sigma(t))_{t\in\mathbb{N}} = \text{Orb}(\Delta, \xi, \sigma)$ obtained with $\sigma(0) = \sigma$ and $\forall\, t \in \mathbb{N}$, $\sigma(t+1) = \Gamma_{\Delta,\xi}(\sigma(t))$. In this paper, we study three updating methods and three conflict resolution policies. These functions are now presented with both informal and formal definitions.

The updating method is a function $\Delta : \mathbb{N} \to \mathcal{P}(T)$, where $\mathcal{P}(X)$ denotes the power set of a finite set X and Δ selects a set of turmites to update at each time step. Three different updating methods Δ are used.

- *Synchronous* update Δ_s: turmites are simultaneously updated at each time step.
- *Cyclic* update Δ_c: turmites are updated sequentially in a fixed order.
- *Random* update Δ_r: turmites are updated sequentially but the order of the updating within each cycle varies randomly.

These updating methods are formally defined with $\forall\, t \in \mathbb{N}$:

$$\Delta_s(t) = \text{T},$$
$$\Delta_c(t) = \{t \bmod N + 1\}, \text{ and}$$
$$\Delta_r(t) = \pi_k(t'),$$

where $t' = t \bmod N$, $k = \lfloor t / N \rfloor$, and $(\pi_k)_{k\in\mathbb{N}}$ is a series of independent random variables that draw a single element uniformly in the set of all permutations of T.

Now that we have defined Δ, let us define Γ by specifying, independently, **S** on the one hand and **P** and **O** on the other. At each time step, the grid state evolves as follows:

$$\forall\, c \in \mathcal{L},\, S_c(t+1) = \begin{cases} 1 - S_c(t) & \text{if } c \in \{P_i,\, i \in \Delta(t)\}, \\ S_c(t) & \text{otherwise.} \end{cases}$$

This rule means that $S_c(t)$, the state of a cell c at time t, is changed if the cell c is selected by Δ and contains at least one turmite. Other policies are possible, for instance, considering the so-called annihilation policy where simultaneous flips are combined by pairs [14].

Let us now describe how to update the positions and orientations of the turmites. Defining $(\mathbf{P}, \mathbf{O})(t)$ requires specifying how turmites interact. Note that if a turmite is not updated, its position and orientation do not change: $\forall\, i \notin \Delta(t)$, $(P_i, O_i)(t+1) = (P_i, O_i)(t)$. For an updated turmite $i \in \Delta(t)$, the way to calculate $(P_i, O_i)(t+1)$ depends on the conflict resolution policy ξ. For a turmite i, starting from its orientation and position at time t, we denote by $\tilde{O}_i(t)$ and $\tilde{P}_i(t)$ the new orientation and position of this turmite without taking into account conflicts with other turmites.

Let us denote by R and L the right and left rotations, respectively, such that $\mathtt{R}(\mathtt{N}) = \mathtt{E}$, $\mathtt{L}(\mathtt{N}) = \mathtt{W}$, ... The functions $\tilde{O}_i(t)$ and $\tilde{P}_i(t)$ are defined by:

$$\tilde{O}_i(t) = \begin{cases} \mathtt{R}(O_i(t)) & \text{if } S_{P_i(t)}(t) = 0 \\ \mathtt{L}\,(O_i(t)) & \text{if } S_{P_i(t)}(t) = 1 \end{cases}$$

and

$$\forall\, i \in T,\, \tilde{P}_i(t) = P_i(t) + \mathrm{dP}\big(\tilde{O}_i(t)\big)$$

where $\mathrm{dP}(\mathtt{N}) = (0, 1)$, $\mathrm{dP}(\mathtt{E}) = (1, 0)$, $\mathrm{dP}(\mathtt{S}) = (0, -1)$, $\mathrm{dP}(\mathtt{W}) = (-1, 0)$ (north, east, south, and west translations). We are now in position to define the three different conflict resolution policies ξ: allow, exclude, and turn and see.

Allow policy (ξ_{Al}). This policy allows turmites to move and rotate freely without taking into account conflicts. Formally,

$$\xi_{\mathrm{Al}} : \forall\, i \in \Delta(t),\ (P_i,\, O_i)\,(t+1) = \big(\tilde{P}_i,\, \tilde{O}_i\big)\,(t).$$

Exclude policy (ξ_{Ex}). When a conflict occurs, the ξ_{Ex} policy prohibits turmite movements and rotations. Conflicts occur in two cases.

- *Type A*. A turmite asks to move to an occupied cell.
- *Type B*. Two or more turmites ask to move to the same target cell.

To express these conflicts formally, we use a function n that counts the number of turmites present in a cell $c \in \mathcal{L}$ given a set of turmite positions $\mathbf{P}$:

$$\begin{aligned} n :\ & \mathcal{L}^N \times \mathcal{L} \to \mathbb{N} \\ & (\mathbf{P},\, c) \mapsto \mathrm{card}\,\{i \in \mathrm{T},\, P_i = c\}. \end{aligned}$$

Figure 2 shows example type A and B conflicts. The positions and orientations of turmites 1 and 2 represent a type B conflict when they both ask to move onto the same cell c_1. The positions and orientations of turmites 3 and 4 represent a type A conflict when turmite 3 asks to move onto the cell c_2 that contains turmite 4.

Formally, we say that a turmite i is in a type A conflict if $n\big[\mathbf{P},\, \tilde{P}_i\big] \neq 0$. Similarly, a turmite i is in a type B conflict if: $\exists\, j$, $i \neq j$, $\tilde{P}_i = \tilde{P}_j$, and $n\big[\mathbf{P},\, \tilde{P}_i\big] = 0$. The exclude policy is then written as:

$$\xi_{\mathrm{Ex}} : (P_i,\, O_i)\,(t+1) = \begin{cases} \big(\tilde{P}_i,\, \tilde{O}_i\big)\,(t) & \text{if turmite } i \text{ is not in conflict,} \\ (P_i,\, O_i)\,(t) & \text{otherwise.} \end{cases}$$

Turn and see policy (ξ_{Ts}). This policy is somewhat of an intermediary policy between the allow and exclude policies. The turmites in-

https://doi.org/10.25088/ComplexSystems.21.3.165

volved in a conflict do not move, but they are allowed to turn. Using the previous notations, this is written as:

$$\xi_{Ts}: \begin{cases} P_i(t+1) = \begin{cases} \tilde{P}_i(t) & \text{if turmite } i \text{ is not in conflict,} \\ P_i(t) & \text{otherwise,} \end{cases} \\ O_i(t+1) = \tilde{O}_i(t). \end{cases}$$

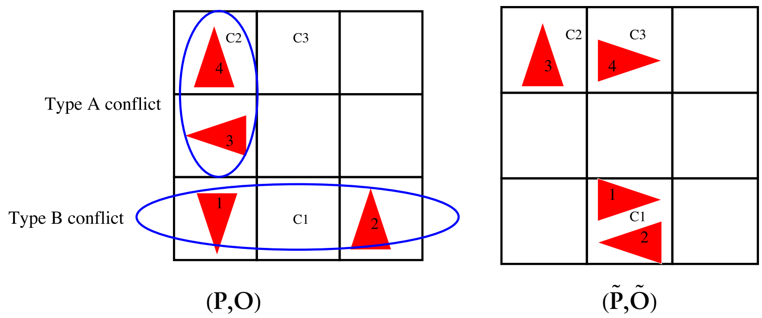


Figure 2. From the current position and orientation $(\mathbf{P}, \mathbf{O})$ of the turmites to their expected next position and orientation $(\tilde{\mathbf{P}}, \tilde{\mathbf{O}})$. We have: $n[\mathbf{P}, c_1] = 0$, $n[\tilde{\mathbf{P}}, c_1] = 2$, $n[\mathbf{P}, c_2] = 1$, $n[\tilde{\mathbf{P}}, c_2] = 1$, $n[\mathbf{P}, c_3] = 0$, and $n[\tilde{\mathbf{P}}, c_3] = 1$. This situation generates a type A conflict with turmite 3, a type B conflict with turmites 1 and 2, and no conflict with turmite 4.

In short, we have three updating methods and three conflict resolution methods, which define nine possible combinations and thus nine dynamical systems Γ that we call *submodels*, since they derive from one single general simulation model. Having multiple submodels of simulation schemes does not prove by itself the importance of formalizing simulation schemes. This importance can only be estimated through its effects, that is, if it qualitatively modifies the orbits. We can now investigate which are the interesting collective phenomena that can be observed with the different submodels.

3. Observations of Collective Phenomena and Their Robustness

Now that we transformed our system from an informal individual-based description to a dynamical systems description, let us observe the perspectives opened by this change of viewpoint. As a first step, we focus our attention on collective phenomena. As an exhaustive exploration of these phenomena is out of reach, it is necessary to select a few phenomena to study. In this section, we select collective phenomena that, in our view, cannot be predicted simply by looking at

https://doi.org/10.25088/ComplexSystems.21.3.165

the local rules that define the system. We are interested in studying these phenomena and also want to know how they are affected by changes in the simulation schemes (updating method and conflict resolution policies).

3.1 Cycles and Clocks

The first phenomenon that caught the attention of researchers was that of a mono-turmite system starting from an all-0 grid that leads to constructing a *path* structure, that is, a translating orbit in which the trace of the turmite has a regular pattern. It may then be wondered whether a single turmite might construct other regular structures. For example, it is known that it is impossible to observe a mono-turmite whose behavior is cyclic [3, 11].

Interestingly, for the multi-turmite system and for particular initial conditions, it is possible to observe cycles. First observations of cycles were reported in the experimental study by Chevrier and Fatès [19]. However, it is in general difficult to predict the form of a cycle as a function of the initial condition. We now present a particular set of initial conditions for which this prediction is possible, forming a phenomenon that we call a *clock*.

Definition 1. (Cycle) An orbit $(\sigma(t))_{t \in \mathbb{N}}$ is a cycle if

$$\exists \ t_0, p \in \mathbb{N}, \ \forall \ t \geq t_0, \ \sigma(t+p) = \sigma(t).$$

The smallest t_0 and p for which the cyclic property is verified are called the *transient time* and the *period* of a cycle, respectively.

Observation 1. For the synchronous allow submodel ξ_{Al}, an even number of turmites placed horizontally next to each other with a north orientation produces a cycle. Formally: for $N \in 2\mathbb{N}$, and for an initial condition $\sigma = (\mathbf{S}, \mathbf{P}, \mathbf{O}) : \mathbf{S} = \mathbf{0}; \ \forall \ i \in T, \ P_i = (i-1, 0), O_i = \mathtt{N}$, the orbit $\text{Orb}(\Delta_s, \xi_{\text{Al}}, \sigma)$ is a cycle.

We experimentally determined that the period of the cycle varies as $16N - 4$ and that the transient time is 0. Moreover, the sets of cells visited by the turmites can be enclosed in a rectangular zone of $3N \times 2N$ cells.

Figure 3 shows a cycle with four turmites. The configuration at time $t = 60$ shows the end of the cycle, that is, when the configuration is identical to the initial configuration.

Robustness

For the set of initial conditions as described, the clock phenomenon is only observed with the synchronous update and the allow policy ($\Gamma_{\Delta_s, \xi_{\text{Al}}}$ submodel; see Table 1). For the eight other submodels considered, no regularity of behavior was observed, at least for the first few hundred steps of evolution. The divergence between the evolution of the different submodels is observed only after a few steps.

https://doi.org/10.25088/ComplexSystems.21.3.165

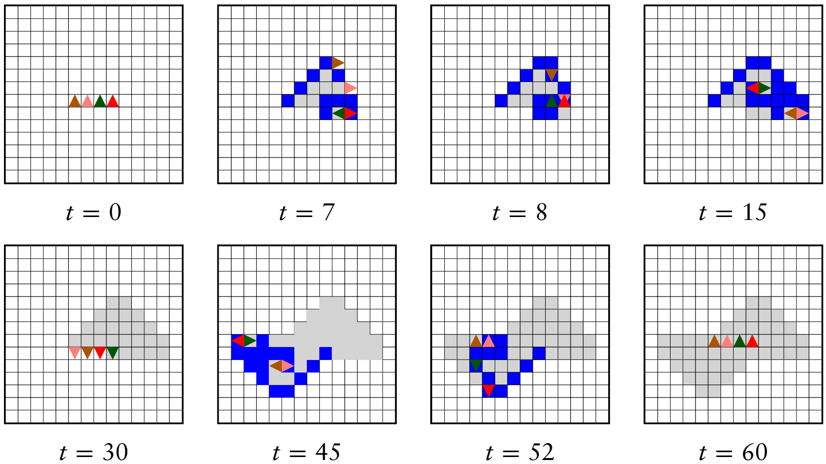


Figure 3. Clock cycle with four turmites. Each turmite has its own color (convention kept).

Phenomena	Submodels	Conflicts
clock	$\Gamma_{\Delta_s}, \xi_{\mathrm{Al}}$	break
B-glider	$\Gamma_{\Delta_s}, \xi_{\mathrm{Al}}$ $\Gamma_{\Delta_c}, \xi_{\mathrm{Al}}$ $\Gamma_{\Delta_r}, \xi_{\mathrm{Al}}$	reversion
stalemate	$\Gamma_{\Delta_s}, \xi_{\mathrm{Ts}}$	dual lock

Table 1. Observation of the three phenomena and the three conflict forms that occur in their evolution. Only submodels that produce a phenomenon appear in the table.

3.2 Gliders

Gliders are rare phenomena in the multi-turmite system; it was necessary to test for thousands of different configurations to observe them. They are "purely translating" patterns, where the turmites go in a straight direction and leave traces with cells in state 0.

Definition 2. (Gliders) We say that an orbit $(\sigma(t))_{t \in \mathbb{N}}$ is a glider if:

- all the turmites are in the same infinite translation, that is, for all $i \in T$:
  $\exists\, \rho \in \mathcal{L}, \exists\, t_0 \in \mathbb{N}, \forall\, t \in \mathbb{N}$,
  $P_i(t + t_0) = P_i(t) + \rho$, and $O_i(t + t_0) = O_i(t)$; and
- starting from an empty grid, the "trace" left by the turmites is 0, that is:
  $S = 0$ and $\exists\, \tau, \forall\, t, \forall\, i \in \mathrm{T}, \; S_{P_i(t)}(t + \tau) = 0\,.$

These gliders are phenomena that are analogous to the gliders observed in the Game of Life cellular automaton [20]. The first glider was the F-glider [19]; we now present a description of the B-glider.

Observation 2. For the allow policy ξ_{Al}, four turmites placed in a square position with a north orientation form a glider. Formally: for $N = 4$, for $\Delta \in \{\Delta_s, \Delta_c, \Delta_r\}$, the orbit $\mathrm{Orb}(\Delta, \xi_{\mathrm{Al}}, \sigma)$ obtained with the initial condition

$$\sigma = (\mathbf{S}, \mathbf{P}, \mathbf{O}) : \mathbf{S} = 0;$$
$$P_0 = (0, 0), P_1 = (1, 0), P_2 = (1, 1), P_3 = (0, 1);$$
$$O_1 = O_2 = O_3 = O_4 = \mathtt{N}$$

is a glider.

Figure 4 shows four steps in the evolution of the B-glider; this glider translates with a distance of two cells (horizontally or vertically) every 12 steps.

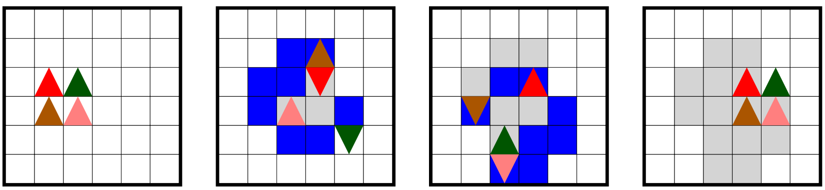


Figure 4. Four steps in the translation cycle of the B-glider.

Robustness

The B-glider is robust to changes in the updating method. However, it is not robust to changes in the conflict resolution policy (see Table 1). Note that, by contrast, the F-glider is only observed with synchronous updating and the allow policy ξ_{Al}.

3.3 Stalemate

We now present our third class of initial conditions, which produces a phenomenon that we call a *stalemate* (originally called a deadlock).

Definition 3. (Stalemate) An orbit $(\sigma(t))_{t \in \mathbb{N}}$ is a stalemate if

$$\exists\, t_0, \forall\, t \geq t_0, \forall\, i \in T,\ P_i(t) = P_i(t_0)\,.$$

The smallest t_0 for which the property is verified is called the *stalemate time*.

Observation 3. For the synchronous turn and see submodel ξ_{Ts}, two turmites placed at a horizontal distance k of each other with orthogonal

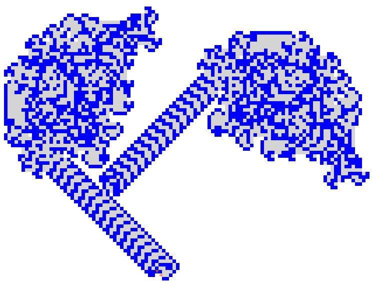
https://doi.org/10.25088/ComplexSystems.21.3.165

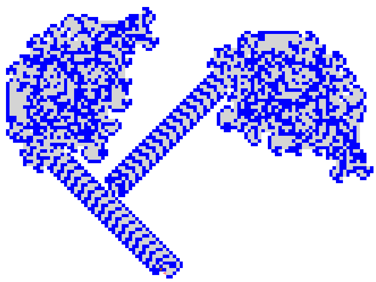
orientations west and north always produce a stalemate if k is a multiple of 4 and greater than 52. Formally: for $N = 2$, for $k \in 4\mathbb{N}^*$ and $k \geq 52$ and

$$\sigma = (\mathbf{S}, \mathbf{P}, \mathbf{O}) : \mathbf{S} = 0;$$
$$P_1 = (0, 0), O_1 = \mathtt{W};$$
$$P_2 = (k, 0), O_2 = \mathtt{N},$$

the orbit $\text{Orb}(\Delta_s, \xi_{Ts}, \sigma)$ is a stalemate.

Figure 5 shows two stalemates obtained with two distances k. Note that although stalemates are also observed for $k < 52$, there is no regularity in the way the phenomenon happens. By contrast, for $k \geq 52$, we can guarantee that the two turmites always interact in the same way. Indeed, this "minimal security distance" is given to ensure that each turmite does generate its own path and is not "perturbed" by an overlap in its trace and the trace of another turmite. If this condition is met, turmite 2 always crosses the path of turmite 1 the same way: it follows the path, turns around it, and then catches up with turmite 1. This produces a type B conflict, which always results in a stalemate.





$k = 52, \ t = 11539$ $k = 56, \ t = 11643$

Figure 5. Two stalemate situations obtained with initial distances of 52 and 56 spaces between the two turmites (see text for a precise description of the initial condition).

Increasing the value of k by four cells ensures that the two turmites follow the same sequence of behavior before they meet. This regularity is due to the spatial periodicity of the path construction, which is invariant by a translation of $(2, -2)$ (see Figure 5). We also observe that an increase of k by four increases the stalemate time by 104, a result in agreement with the analytical results obtained by Boon [7]. The stalemate time thus varies with k as:

$$t_{st}(k) = t_{st}(52) + \left(\frac{k - 52}{4}\right) \times 104,$$

with the experimental value $t_{st}(52) = 11\,539$.

https://doi.org/10.25088/ComplexSystems.21.3.165

Robustness

Stalemates were observed only with the synchronous turn and see submodel ($\Gamma_{\Delta_s, \xi_{Ts}}$; see Table 1). The sensitivity of this phenomenon to asynchrony is analyzed in Section 4.

To sum up, we described three phenomena that result from the collaboration of turmites. We observed that, depending on the phenomenon considered, there exist a wide variety of system responses to various updating schemes and conflict resolution policies. We now endeavor to explain some of these variations of behavior by means of microscopic analysis.

4. Microscopic Analysis of the Robustness

To understand how the variation from one submodel to another affects the global behavior of the system, our method consists in examining the evolution of the system until we identify the time steps where a spatial conflict appears. We then try to establish a relationship between the type of collision and the robustness of the system from a global point of view.

4.1 Sensitivity of the Clock

Recall that the "clock" is a collaborative phenomenon that was observed only with the synchronous allow policy ξ_{Al}. Let us explain this sensitivity to asynchrony and to the changes of conflict resolution policy.

By observing the evolution of the initial condition that generates a clock with the synchronous allow policy ξ_{Al}, we remarked that:

- If the turn and see ξ_{Ts} or exclude ξ_{Ex} policies are used, a divergence with the allow policy ξ_{Al} appears after only one step since the movements of all the turmites but the rightmost one are blocked by these two policies.
- When using an asynchronous allow policy ξ_{Al}, the divergence with synchrony appears later, at a time that depends on the number of turmites involved. For instance, for four turmites, it appears after eight steps.

To identify the origin of the sensitivity, we observed the orbit of the clock and noticed that it contains a particular type of spatial conflict that we call the *break* conflict. This conflict is characterized by a particular configuration where two turmites are sharing the same target cell and have the same direction. It is visible in Figure 3 at time $t = 8$.

As can be seen in Figure 6 ($t = 1$), when a break conflict appears, the two turmites leave the cell with opposite directions in the synchronous mode while they leave it with identical directions in the asynchronous mode. The presence of this conflict thus explains the

https://doi.org/10.25088/ComplexSystems.21.3.165

sensitivity of this pattern to variations of temporal updates: a local divergence in trajectories appears and this small divergence is amplified until it generates qualitatively different orbits (see Table 1). Interestingly, it is also the existence of a similar conflict that explains the sensitivity of the F-glider. Let us now examine a pattern whose response to asynchronism is radically different.

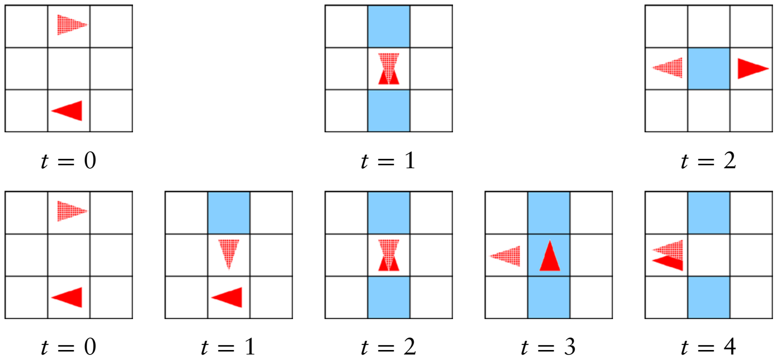


Figure 6. Microscopic analysis showing the fragility of the break conflict (cells in state 1 are blue). Allow policy ξ_{Al} with: (top) synchronous and (bottom) asynchronous updates. The hatched turmite is updated before the plain turmite (convention kept in the following figures).

4.2 B-Glider

Contrarily to the F-glider, the B-glider is robust to changes in the updating scheme, but not to changes in the spatial conflict policy. An analysis of its different steps of evolution shows that only one form of conflict is involved. This conflict appears twice during the cycle: in Figure 4, it is at times $t = 4$ and $t = 8$.

We call this particular form of conflict the *reversion* conflict. For two turmites i and j, it is characterized by the pattern represented in Figure 7: $O_i = O_j$, $P_j = P_i + (1, 0)$, $S_i = 0$, $S_j = 1$ (the other patterns obtained by translations and 90° rotations are of course equivalent).

As this is a particular case of type A conflict, the allow policy ξ_{Al} lets the turmites swap their positions. With the asynchronous update, the exchange happens in two steps but the result is the same as in the synchronous case, whatever the updating order of the turmites (see Figure 7). This similarity of evolution explains the robustness of the glider to the asynchronous update.

On the other hand, the turn and see ξ_{Ts} and exclude ξ_{Ex} submodels yield different behaviors since type A conflicts imply a divergence in the evolution of the turmites: their positions are not modified, but their cell state is. Remarkably, this conflict generally leads to the production of cyclic or translating orbits. Indeed, when it appears, its ef-

https://doi.org/10.25088/ComplexSystems.21.3.165

fect results in the inversion of the roles of the two turmites. Once the conflict has occurred, we generally observe that each turmite erases the trace left by the other turmite. This phenomenon has already been observed by other authors (e.g., [11, 19]) but was only partly explained.

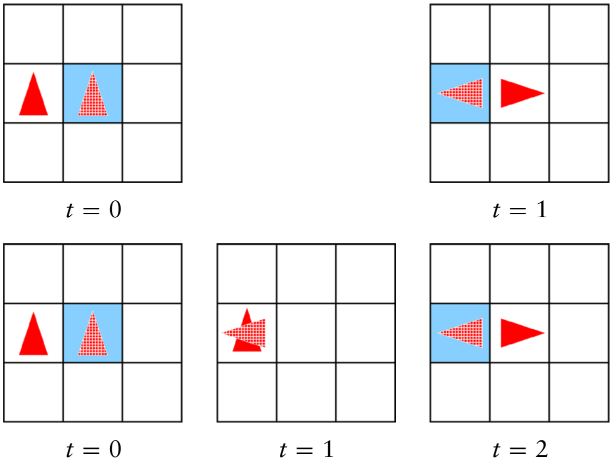


Figure 7. Microscopic analysis showing the robustness of the reversion conflict. Allow policy with: (top) synchronous update and (bottom) asynchronous update.

4.3 Stalemates

The occurrence of stalemates is characterized by a situation where turmites are always in a type B conflict with perpendicular orientations. We call this type of conflict the *dual lock pattern* (see Figure 8). Recall that stalemates were observed only with the turn and see policy using the synchronous update.

To explain why, let us consider two turmites i and j such that: $P_j = P_i + (1, -1)$, $O_i = \texttt{E}$, $O_j = \texttt{N}$, $S_i = 0$, $S_j = 1$. As a type B conflict occurs on the cell $c = \tilde{P}_i = \tilde{P}_j$, the new orientations and positions are: $O'_i = \texttt{S}$, $O'_j = \texttt{W}$ and $P'_i = P_i$, $P'_j = P_j$ (the positions of the turmites do not change). Then, the same type of conflict appears on the cell $c' = \tilde{P}'_i = \tilde{P}'_j$ and the turmites are again in a type B conflict that results in a stalemate.

On the contrary, with an asynchronous updating, turmite i moves before turmite j, which frees the stalemate (see Figure 8).

https://doi.org/10.25088/ComplexSystems.21.3.165

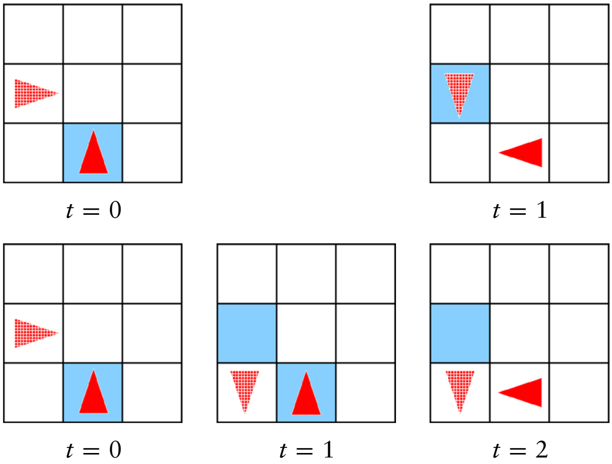


Figure 8. Microscopic analysis of the dual lock conflict. Turn and see policy with: (top) synchronous update and (bottom) asynchronous update.

Clearly, no stalemate can occur with an allow policy ξ_{Al} as turmites are not blocked by conflicts. With the exclude policy ξ_{Ex}, turmites keep the same orientation and position but the state of their cell changes. This brings them to move in the opposite direction and solves the conflict.

The dual lock conflict thus explains why the stalemate phenomenon was observed only with the synchronous turn and see submodel ξ_{Ts} (results reported in Table 1). In general, we observed that orbits that involve type B conflicts are not robust to the asynchronous updating.

5. Discussion

We presented clocks, gliders, and stalemates as three emergent phenomena in a multi-agent system composed of turmites that evolve on an infinite square grid. Their robustness was tested with nine different simulation schemes. These simulation schemes were defined with a dynamical systems approach using a combination of updating methods and conflict resolution policies. This allowed us to have an unambiguous description of the interactions in the system, a criterion necessary for reproducing the experiments.

The formalism we employed allowed us to define various types of orbits and thus to give a rigorous—although partial—definition of the robustness of those phenomena to asynchrony. We exhibited a correlation between the robustness of the orbits and the conflicts that occurred during turmite movements. This correlation was explained

https://doi.org/10.25088/ComplexSystems.21.3.165

with a microscopic analysis of the conflicts (see Table 1 for a synthesis).

Although the classical Langton's ant (mono-turmite) system is still not fully understood, the multi-turmite system opens an even wider realm to discover. Many other puzzling phenomena deserve to be studied, for instance, the so-called ever-growing square (or diamond) where the turmites collaborate to produce a pattern that progressively expands [1, 19]. The possibility of using turmites to generate textures is another direction of research.

The relationship that we noticed between the initial condition, the conflicts' forms, and the global behavior of the system all deserve further analysis. A challenging problem is to derive stronger relationships to predict the robustness of a phenomenon given the conflicts involved, for instance, with a proper analysis of the initial condition (e.g., symmetries or conserved quantities). We already know from the work of Gajardo et al. that a single-agent system is Turing universal [8]. Is the sensitivity of this system to simulation conditions related to its computational universality? In the case of the Game of Life, it was observed that the system is not robust to asynchronous updating [20]; can we find particular constructions of the multi-turmite system that would show some robustness to multiple variations of its simulation scheme? We believe that a deeper understanding of clocks, gliders, and stalemates as well as other collaborative phenomena may provide some hints to answer these questions.

Finally, we ask how the investigations made on the robustness of the multi-turmite system can be related to other types of complex systems. In particular, it would be interesting to compare the robustness of our model with the lattice-gas cellular automata models of Chopard and Droz [11] or to other models where the interactions between cells and turmites have a physical interpretation. As the qualitative behavior of a system may highly depend on small simulation details, we need to develop specific tools to understand how interactions generate robust or sensitive collaborative phenomena. Is the sensitivity to the simulation scheme observed here a rare or a common phenomenon? Can it be seen only in simple, discrete, and deterministic models or can it also be observed in a wider range of models?

Acknowledgments

Selma Belgacem gratefully acknowledges the support of the LORIA laboratory, the Inria institute, and the MaIA team where this research was conducted as a part of her master's thesis.

References

[1] C. G. Langton, "Studying Artificial Life with Cellular Automata," *Physica D: Nonlinear Phenomena*, **22**(1–3), 1986 pp. 120–149. doi:10.1016/0167-2789(86)90237-X.

[2] A. M. Turing, "The Chemical Basis of Morphogenesis," *Philosophical Transactions of the Royal Society of London B*, **237**(641), 1952 pp. 37–72. doi:10.1098/rstb.1952.0012.

[3] L. A. Bunimovich and S. E. Troubetzkoy, "Recurrence Properties of Lorentz Lattice Gas Cellular Automata," *Journal of Statistical Physics*, **67**(1–2), 1992 pp. 289–302. doi:10.1007/BF01049035.

[4] A. K. Dewdney, "Computer Recreations: Two-Dimensional Turing Machines and Tur-mites Make Tracks on a Plane," *Scientific American*, **261**, 1989 pp. 180–183. doi:10.1038/scientificamerican0989-180.

[5] A. Khuong, G. Theraulaz, C. Jost, A. Perna, and J. Gautrais, "A Computational Model of Ant Nest Morphogenesis," in *Advances in Artificial Life, ECAL 2011: Synthesis and Simulation of Living Systems* (T. Lenaerts, M. Giacobini, H. Bersini, P. Bourgine, M. Dorigo, and R. Doursat, eds.), Cambridge, MA: MIT Press, 2011 pp. 404–411.

[6] D. Gale, J. Propp, S. Sutherland, and S. Troubetzkoy, "Mathematical Entertainments: Further Travels with My Ant," *Mathematical Intelligencer*, **17**(3), 1995 pp. 48–56. doi:10.1007/BF03024370.

[7] J. P. Boon, "How Fast Does Langton's Ant Move?" *Journal of Statistical Physics*, **102**(1–2), 2001 pp. 355–360. doi:10.1023/A:1026581213671.

[8] A. Gajardo, A. Moreira, and E. Goles, "Complexity of Langton's Ant," *Discrete Applied Mathematics*, **117**(1–3), 2002 pp. 41–50. doi:10.1016/S0166-218X(00)00334-6.

[9] A. Gajardo, E. Goles, and A. Moreira, "Generalized Langton's Ant: Dynamical Behavior and Complexity," *Lecture Notes in Computer Science*, **2001**, 2010 pp. 259–270. doi:10.1007/3-540-44693-1_23.

[10] A. Gajardo and E. Goles, "Dynamics of a Class of Ants on a One-Dimensional Lattice," *Theoretical Computer Science*, **322**(2), 2004 pp. 267–283. doi:10.1016/j.tcs.2004.03.012.

[11] B. Chopard and M. Droz, *Cellular Automata Modeling of Physical Systems*, Cambridge, UK: Cambridge University Press, 1998 pp. 46–51.

[12] O. Beuret and M. Tomassini, "Behaviour of Multiple Generalized Langton's Ants," in *Proceeding of the Fifth International Workshop on the Synthesis and Simulation of Living Systems* (C. Langton and K. Shimohara, eds.), Nara, Japan, Cambridge, MA: MIT Press, 1998 pp. 45–50.

[13] J. Ferber and J. P. Müller, "Influences and Reaction: A Model of Situated Multiagent Systems," in *Proceedings of the Second International Conference on Multiagent Systems*, Kyoto, Japan, Menlo Park, CA: AAAI Press, 1996 pp. 72–79.

[14] V. Chevrier and N. Fatès, *Multi-agent Systems as Discrete Dynamical Systems: Influences and Reactions as a Modelling Principle*, Research report inria-00345954, INRIA - LORIA, 2008. http://hal.inria.fr/inria-00345954/en.

https://doi.org/10.25088/ComplexSystems.21.3.165

[15] R. L. Buvel and T. E. Ingerson, "Structure in Asynchronous Cellular Automata," *Physica D: Nonlinear Phenomena*, **10**(1–2), 1984 pp. 59–68. doi:10.1016/0167-2789(84)90249-5.

[16] N. Fatès and L. Gerin, "Examples of Fast and Slow Convergence of 2D Asynchronous Cellular Systems," *Journal of Cellular Automata*, **4**(4), 2009 pp. 323–337.

[17] D. Regnault, N. Schabanel, and E. Thierry, "Progresses in the Analysis of Stochastic 2D Cellular Automata: A Study of Asynchronous 2D Minority," *Theoretical Computer Science*, **410**(47–49), 2009 pp. 4844–4855. doi:10.1016/j.tcs.2009.06.024.

[18] N. Fatès, "Does *Life* Resist Asynchrony?" in *Game of Life Cellular Automata* (A. Adamatzky, ed.), London: Springer, 2010 pp. 257–274.

[19] V. Chevrier and N. Fatès, "How Important Are Updating Schemes in Multi-agent Systems? An Illustration on a Multi-turmite Model," in *Proceedings of the Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Toronto, Canada, 2010 pp. 533–540.

[20] E. R. Berlekamp, J. H. Conway, and R. K. Guy, *Winning Ways for Your Mathematical Plays, Volume 2: Games in Particular*, London: Academic Press, 1982, chapter 25.