Chapter 12
Modelling Synchronisation in Multirobot Systems with Cellular Automata: Analysis of Update Methods and Topology Perturbations

Fernando Silva, Luís Correia and Anders Lyhne Christensen

Abstract In this chapter, we consider two-dimensional cellular automata as a tool for modelling the behaviour of multirobot systems. We study the dynamics of a group of stationary robots inspired by studies in mixed natural-artificial societies. We model the behaviour of individual robots as a pulse-coupled oscillator, which influences other oscillators through short and periodic pulses. We address the problem of self-organised synchronisation, in which robots have to adjust and synchronise their behaviour to produce a self-organised collective vibration pattern based on local interactions. We analyse under which conditions a synchronised global behaviour can emerge from local coupling between neighbours and focus on two fundamental aspects: (i) the effects of different update methods, including the interplay between parameters of local rules and the global behaviour, and (ii) the transition from regular to irregular topologies by means of dynamic perturbations and fixed perturbations. Overall, this study is a contribution towards the understanding of the effects of the update method and of the topological structure when modelling real-world complex systems with cellular automata.

12.1 Introduction

In this chapter, we study two-dimensional cellular automata as an abstraction for modelling and examining the behaviour of autonomous, mutually interacting sets of agents, i.e., multiagent systems. The key principles for modelling multiagent systems by means of discrete dynamical systems such as cellular automata have not

F. Silva (✉) · L. Correia
LabMAg, Faculdade de Ciências da Universidade de Lisboa, 1749-016 Lisbon, Portugal
e-mail: fsilva@di.fc.ul.pt

L. Correia
e-mail: luis.correia@di.fc.ul.pt

A.L. Christensen
Instituto de Telecomunicações, Instituto Universitário de Lisboa (ISCTE-IUL),
1649-026 Lisbon, Portugal
e-mail: anders.christensen@iscte.pt

G.Ch. Sirakoulis and A. Adamatzky (eds.), *Robots and Lattice Automata*,
Emergence, Complexity and Computation 13, DOI 10.1007/978-3-319-10924-4_12

yet been agreed upon [1]. Proposed approaches include: (i) translation of multiagent systems into corresponding cellular automata models [2], (ii) modification of the expressiveness and structure of cellular automata in order to provide a basis for direct modelling of groups of interacting agents [3], and (iii) hybrid approaches encompassing both cell-based updating and agent-based updating [4]. Influenced by the view of multiagent systems as discrete dynamical systems provided by Fatès and Chevrier [5], we define and model a multiagent system by separating three core concepts: (i) the formulation of individual behaviour, i.e., the local rules that describe the actions of the agents, (ii) the update method, which defines the temporal relation between modifications to the agents' internal state, and (iii) the topology of the environment, including connections between agents, which may change or remain fixed throughout time.

We study the collective dynamics of one class of multiagent systems: multirobot systems. The robots are modelled after the combined actuator-sensor units (CASUs), groups of stationary robots developed in the context of the ASSISIbf project [6] on mixed natural-artificial societies.[1] CASUs are designed according to the principles of distributed control and self-organisation. The robots are expected to coexist and interact with honeybees and zebrafish both at the individual level and at the group level. The CASUs are intended: (i) to modulate the behaviour of the animals, and (ii) to "learn the social language of the animals" [6].

In distributed systems, synchronisation plays a fundamental role [7]. Examples are ubiquitous and include: (i) imposing a reference timing in wireless networks [8], (ii) communication scheduling, coordinated duty cycling, and time synchronisation in sensor networks [9], and (iii) decentralised fault detection in groups of autonomous robots [10]. In our study, robots have to synchronise to produce self-organised collective behaviour. The robots are equipped with actuators to emit vibrations, and sensors that enable them to detect if neighbouring robots are vibrating. The behaviour of each robot is modelled as a pulse-coupled oscillator [11], which influences other oscillators through short and periodic pulses. Based on local interactions, robots have to adjust their behaviour in order to produce a common, population-wide vibration pattern. Our goal is to analyse and understand under which conditions a synchronised global behaviour can emerge from local coupling between neighbours. Our study is concerned with answering three fundamental questions:

- Since robots are not centrally synchronised, does the global self-synchronised behaviour of the multirobot system depend on the update method? Is the system sensitive or robust against changes in the updating? What is the most appropriate update method for modelling multirobot synchronisation of behaviour?
- Under a given update method, is the global behaviour influenced by variations of the parameters that regulate the individual behaviour of agents? In other words,

[1] ASSISIbf [6], short for "Animal and robot Societies Self-organise and Integrate by Social Interaction", is an EU-funded FP7 FET project. The project focuses on self-organising mixed societies of real social animals, namely bees and fish, and artificial systems. The ultimate goal of the project is to develop a new field of science concerning self-adapting engineered systems able to integrate themselves in existing natural societies.

what is the main cause of emergence of certain classes of behaviour: (i) specific configurations of parameters for the local rules, (ii) the update method, or (iii) the interplay between the configuration of the local rule parameters and the update method?
- How robust is the system to modifications in the topology? In particular, if the topology is perturbed by the removal of connections and therefore interactions between neighbouring agents, under which conditions can the system continue to exhibit the desired collective behaviour?

The chapter is organised as follows. Section 12.2 describes the background and related work. We present the cellular automata model, the update methods used, and we review relevant studies on pulse-coupled oscillators. In Sect. 12.3, we introduce a number of definitions and notations necessary to characterise the behaviour of cellular automata. In Sect. 12.4, we assess the impact of the update method on the evolution of synchronised behaviour. We also analyse the sensitivity of the update method to variations in key parameters of individual behavioural rules. In Sect. 12.5, we investigate the effects of topology perturbations. In particular, we analyse the robustness of cellular automata when topology characteristics become irregular due to the removal of connections between neighbouring cells. In Sect. 12.6, we summarise and discuss our contribution, and we present avenues for future research in modelling multirobot systems with cellular automata.

12.2 Background and Related Work

In this section, we first define the cellular automata model and we describe the five update methods studied in this chapter. Afterwards, we report the background and related work on oscillators, and we detail our pulse-coupled oscillator model.

12.2.1 Topology of the Environment

Let A be a two-dimensional squared automaton of length l with toroidal boundary conditions. We denote $N = l \times l$, the total number of cells, as the size of the automaton. Individual cells adopt one of two possible states in the set of states $S = \{0, 1\}$. The state of all cells at a given time is called a *configuration*.

The state of each cell is updated at discrete time steps according to a local transition function. The locality of the interactions between cells is given by the relation of each cell $c \in A$ with its *von Neumann* neighbourhood of radius $r = 1$: $N(c) = \{c, c + \overrightarrow{v}, c - \overrightarrow{v}, c + \overrightarrow{h}, c - \overrightarrow{h}\}$, where $\overrightarrow{v} = (0, r)$ and $\overrightarrow{h} = (r, 0)$. For a given cell c and an integer value k, we define the *field* $\varphi(c, k)$ as:

$$\varphi(c, k) = \{c' \in A \mid r(c, c') = k\}\ , \tag{12.1}$$

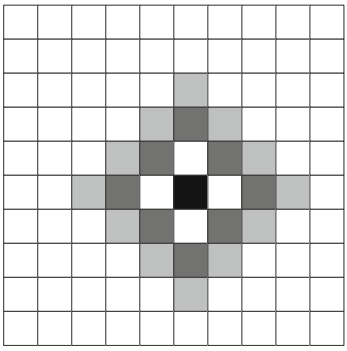

Fig. 12.1 Example of two fields, $\varphi(c, 2)$ and $\varphi(c, 3)$, for an automaton of size $N = 10 \times 10$. The centre cell c is marked in *black*. The fields $\varphi(c, 2)$ and $\varphi(c, 3)$ are marked in *dark grey* and *light grey*, respectively

where $r(c, c')$ is the radial distance between cells c and c'. In Fig. 12.1, we show an example of the fields $\varphi(c, 2)$ and $\varphi(c, 3)$ for an automaton of size $N = 10 \times 10$.

12.2.2 Update Methods

The update method determines the set of cells to which the local transition function applies and the order of the updating at each time step. In the classic definition, cellular automata are perfectly synchronous as cells are updated instantaneously and in parallel. The assumption of perfectly synchronous timing has been widely debated, with the main argument against being that synchrony presupposes the existence of a global clock that dictates the pace of all local processes in the system [12]. In addition, the synchronous update does not reflect the microscopic structure of physical and biological systems [13, 14]. Immediately after the issue of synchrony vs. asynchrony was raised, a number of additional research questions arose [15] including: Which properties of the system change when the update method is asynchronous instead of synchronous? What type of asynchrony should be used to model a given system?

A number of studies focusing on asynchronous update methods have been conducted, see for instance [16–22]. The contributions showed that the behaviour of cellular automata can change considerably when perfect synchrony is absent. Bersini and Detours [23], Ruxton and Saravia [24] and Cornforth et al. [13] argued that no single update method is suitable for all systems. Here, in order to understand which update method is more faithful to the physical reality of a given system, we study the effects of five different schemes:

1. Synchronous method, in which all cells are updated in parallel at each time step.
2. α-asynchronous method [19], which is based on a sampling scheme in which only a fraction of the cells is updated at each time step. Every cell has an independent and equal probability α of being updated, with $0 < \alpha < 1$, thus satisfying a fair sampling condition. The α-asynchronous method has been widely used to: (i) define robustness classes [19] and analyse behavioural changes under

asynchronous conditions in one-dimensional [25–28] and two-dimensional cellular automata [29–31], and (ii) to model multiagent systems in two-dimensional cellular automata [1, 32].

3. κ-scaling method [22], which extends standard α-asynchrony in order to compensate for fewer updates due to increasing asynchrony. The κ-scaling method is defined as follows: given a synchrony rate α, at each time step, $\kappa = 1/\alpha$ updates are performed. For non-integer $\kappa = 1/\alpha$, decimal values are probabilistically used for deciding between performing n or $n + 1$ intermediate updates. The resulting configuration is considered as the automaton's next stage. κ-scaling introduces a time-scaling phenomenon according to the synchrony rate, thus potentially normalising the differences introduced by α-asynchrony. In this way, the sampling compensation allows us to assess if differences in behaviour are due to less cell activity (asynchrony) or to more complex phenomena.
4. Random order asynchronous method in which, at each time step, all cells are updated exactly once according to a random order. Kanada [12], Schönfisch and de Roos [21] and Cornforth et al. [13] have previously assessed the effects of randomised update sequences in cellular automata. Randomness in the computational order was shown to significantly influence the spatiotemporal behavioural patterns. Depending on the local transition function, one-dimensional cellular automata were found to exhibit chaotic behaviour [12] and quasi-cyclic behaviour [13]. The random model update method has also been applied to cellular automata-based modelling of, for instance, biological systems [33] and chemical systems [34].
5. Line-by-line sweep, or fixed directional, is the simplest asynchronous method. All cells are updated at each time setup. The update is performed line-by-line, from the leftmost cell to the rightmost cell. The effects of this method have been compared with others by Rakewsky et al. [35] in the asymmetric simple exclusion process (ASEP), by Schönfisch and de Roos [21] in one-dimensional cellular automata, and by Ruxton and Saravia [24] in two-dimensional cellular automata-based ecological modelling. One of the conclusions of the studies was that the line-by-line sweep can introduce additional structure in the evolution of the automata. We use the method to analyse if such a cyclic behaviour can offer any benefit in modelling synchronisation of multirobot systems.

12.2.3 Modelling Individual Behaviour with Pulse-coupled Oscillators

Our study is concerned with modelling and studying the emergence of synchronised behaviour in a population of *stationary* robots. The behaviour of each individual robot is modelled as a pulse-coupled oscillator [11]. In the following sections, we describe the motivation for the task, the background on pulse-coupled oscillators, the robot model, and how the behavioural control of robots is defined.

12.2.3.1 Synchronisation of Pulse-coupled Oscillators

Self-organised synchronisation is a common and important phenomenon in natural systems. A number of natural collective systems are subject to a strong and regular synchronisation component, and they synchronise their behaviour based on local interactions, i.e., simple coupling rules at the individual level result in a consistently synchronised behaviour. Complete synchronisation of behaviour is thus a progressive and emergent phenomenon [36]. Examples of natural synchronisation phenomena include tropical fireflies that routinely synchronise their rhythmic flashes [37], cardiac cells [38], circadian pacemaker cells in the brain, metabolic synchrony in yeast cell suspensions, and crickets that chirp in unison [39].

The phenomenon of collective synchronisation in a number of natural systems is described by pulse-coupled oscillators that spontaneously lock into a common phase. Each oscillator periodically emits a self-generated pulse. When other oscillators observe the pulse, they slightly shift their own oscillation phase. This process leads to an alignment of phases and to synchronised behaviour. Let us consider the aforementioned tropical fireflies. At an individual level, fireflies have neural timing mechanisms, an oscillator. The only interaction occurs when fireflies flash, which stimulates or inhibits the oscillation frequency of neighbouring fireflies and causes them to modify their internal rhythm [11].

The first example of synchronisation of pulse-coupled oscillators was described by Peskin [40], who observed the phenomenon in the cardiac pacemaker cells. Afterwards, Mirollo and Strogatz [11] demonstrated that any number of pulse-coupled oscillators is always able to synchronise their firing rates as long as: (i) the population of oscillators is fully connected, i.e., each oscillator is coupled with all the others, (ii) oscillators are identical with respect to their dynamics, and (iii) the function governing the evolution of the internal state of oscillators over time is smooth, monotonically increasing, and concave down.

A study by Bottani followed [41], demonstrating that globally-coupled oscillators with pulse interaction can synchronise under broader conditions than those considered in the theorem of Mirollo and Strogatz. More recently, Lucarelli and Wang [42] showed that local nearest neighbour coupling among oscillators is sufficient, even in systems whose topology changes every T_c time units, provided that the entire network is connected every 2 T_c time units. It should be noted that in all studies described above, for the network of oscillators to converge into a self-synchronised state, one key assumption is that all oscillators have the same or nearly identical frequencies.

12.2.3.2 Definition of Pulse-coupled Oscillators

In a population of pulse-coupled oscillators, each individual i maintains an internal activation level x_i that increases over time until a given threshold is reached. Then, the oscillator i fires, x_i is reset to zero, and the cycle repeats. When a neighbour oscillator j observes the firing of oscillator i, it further increases its own activation

level x_j. If x_j exceeds the firing threshold, the oscillator j fires, resets its activation level to zero, and restarts its behaviour.

Analytically, general pulse-coupled networks can be written as [43]:

$$\dot{x_i} = f_i(x_i) + \epsilon \sum_{j=1}^{N} g_{ij}(x_i)\delta(t - t_j^* - \eta_{ij}) \,, \tag{12.2}$$

where x_i denotes the activation level of oscillator i, and the function f_i describes its dynamics. The *coupling constant* ϵ denotes the strength of interactions between oscillators. N is the set of oscillators neighbours of i. The pulse-coupling function g_{ij} describes the effect of the firing of another oscillator j on i. The time t_j^* marks the moment when oscillator j last fired. When j fires, the activation level x_i is increased by $\epsilon g_{ij}(x_i)$ after a delay $\eta_{ij} \geq 0$. The increment is given in the form of the Dirac delta function δ satisfying $\delta(t) = 0$ for all $t \neq 0$, $\delta(0) = \infty$, and $\int \delta = 1$. In Fig. 12.2, we show an example of the interactions between two oscillators i and j.

12.2.3.3 Robot Model and Individual Behavioural Control

In this study, we model the behaviour of individual robots with pulse-coupled oscillators. Each cell in a given automaton corresponds to one stationary robot. We

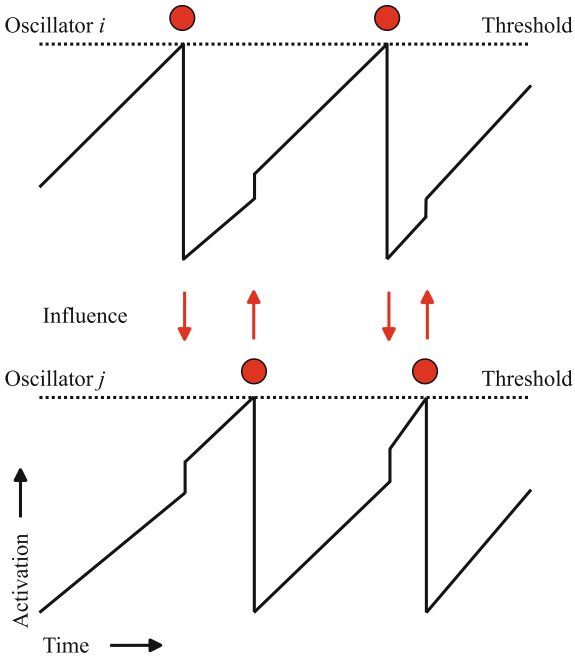

Fig. 12.2 Example of the interactions between two pulse-coupled oscillators i and j. Each oscillator increases its activation level at a constant rate until: (i) the threshold is reached, which resets the activation level to zero, or (ii) until the firing of the other oscillator is detected, at which point the activation of the oscillator that sensed the firing is increased by $\epsilon g(x)$, where ϵ is the coupling constant and $g(x)$ is the pulse-coupling function

model the robots after the combined actuator-sensor units (CASUs), static robotic nodes described by Schmickl et al. [6] in the context of the ASSISIbf project. In our experiments, robots have to collectively adjust and synchronise to produce a local cue, a common vibration pattern. The robotic devices are equipped with: (i) actuators to emit vibrations at a fixed frequency, and (ii) sensors to detect if neighbouring robots are vibrating. Given the discreteness of the cellular automata model, we transform the continuous model for general pulse-coupled systems described in Eq. 12.2 into a discrete time dynamical system with piecewise dynamics, similarly to Christensen et al. [10], as follows:

$$x_i(n+1) = \begin{cases} x_i(n) + \frac{1}{T} + \epsilon\gamma_i(n)g(x_i(n)) & \text{if } x_i(n) < 1 \\ 0 & \text{otherwise} \end{cases} \tag{12.3}$$

where $x_i(n)$ is the activation level of oscillator i at time step n, T is the period between firings of an isolated oscillator, and ϵ is the coupling constant. $\gamma_i(n)$ is the number of neighbours of i that fired in time step n, and $g(x_i(n))$ is the pulse-coupling function. If the activation level of one oscillator exceeds the threshold of 1.0, the corresponding robot vibrates at a fixed frequency, and the state of the cell is set to 1. If the oscillator is not pulsing, the robot is not vibrating and therefore the state of the cell is 0. An example of the development of the activation levels during a run with 100 oscillators is shown in Fig. 12.3. In the example provided, the oscillators synchronise after 1,270 time steps.

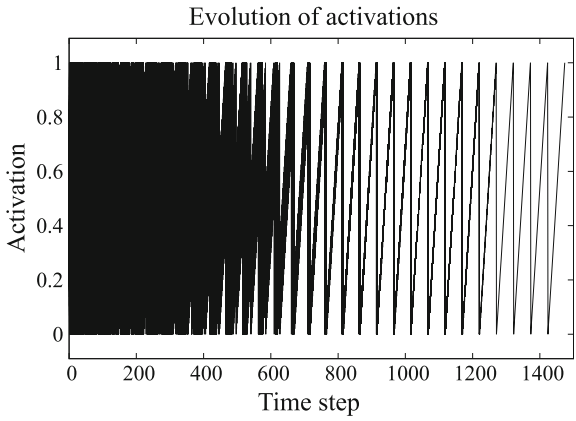

Fig. 12.3 Example of the evolution of activations levels in a population of 100 pulse-coupled oscillators over the course of 1,500 time steps. After time step 1,270, the activation levels overlap, which indicates that the oscillators have synchronised

12.3 Experimental Assessment

In this section, we introduce a number of definitions and notations that we use to analyse the behaviour of cellular automata. In cellular automata, the analytical prediction and classification of behaviour is a complex problem [44]. Depending on the local transition function, aspects such as the update order of the cells, the degree of asynchrony, and the initial configuration can influence the dynamics of the system. In a number of situations, there is even no direct relation between the dynamics of automata following the same local function but subject to different update conditions, see [22, 26, 45] for examples.

12.3.1 Characterising the Behaviour of Cellular Automata

We analyse the behaviour of the automata based both on quantitative and qualitative observations. One of the aspects we are interested in analysing is the time necessary for the system to synchronise under different experimental conditions. We define a cellular automaton of size $N = l \times l$ to exhibit *synchronised* behaviour if: (i) the activation of the first cell is no further than l time steps from all other activations, and (ii) each cell only fires once during this period. Given that each cell only senses the states of its nearest neighbours, our definition of synchronised behaviour takes into account the theoretically *maximum* latency of the system, i.e., the number of time steps necessary for the information to propagate across the automaton.

We estimate the behaviour of different automata and measure the *quality of the synchronisation* process using two methods:

1. We analyse the space-time diagrams of the automata under execution. This form of analysis provides a qualitative impression of behaviour, but does not serve as a formal classification criterion.
2. Complementarily, we quantify the behaviour of the automata using a number of domain-dependent measures, namely:

   - *Convergence rate*, defined as the number of runs in which the automata are able to synchronise their behaviour.
   - *Transient time*, computed as the number of time steps necessary for a given automaton to converge into a synchronised state after it was started.
   - *Speed of collective oscillation*, defined as the total number of time steps necessary for *one instance* of collective synchronised behaviour to be completed, i.e., the time elapsed between the first and the last activation of cells in the automaton.
   - *Period between oscillations*, defined as the number of time steps elapsed *between* different instances of synchronised behaviour. If the automata are properly synchronised, this period should be consistent throughout time, and in accordance with the value T described in Eq. 12.3.

Additionally, cellular automata quantification of behaviour is based on a domain-dependent measure denoted ρ^*. $\rho^* \in [-1, 1]$ measures the degree of correlation between two configurations c_1 and c_2 based on a normalised Hamming distance [46] as follows:

$$\rho^* = 1 - \frac{2 \cdot \sigma_{Ham}(c_1, c_2)}{len(c_1)} , \tag{12.4}$$

$$\sigma_{Ham}(c_1, c_2) = \sum_{i=1}^{len(c_1)} 1 - \delta(c_1[i], c_2[i]) , \tag{12.5}$$

where $\sigma_{Ham}(c_1, c_2)$ is the Hamming distance between c_1 and c_2, $len(c_1) = len(c_2)$ is the length of the configurations, and $\delta(i, j)$ is the Kronecker delta computed as:

$$\delta(i, j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{12.6}$$

$\rho^* = 1$ indicates that c_1 and c_2 have identical values for all cells, and $\rho^* = -1$ means that all cells have complementary states.

ρ^* was used in previous studies in order to: (i) classify the behaviour of one-dimensional cellular automata [47], and (ii) to identify behavioural responses of one-dimensional cellular automata when subject to asynchrony and to noise in the computation of the cell states [22]. In this study, we use ρ^* to quantify cellular automata behaviour in two ways:

1. *Intra-automata correlation*, henceforth intra-ρ^*, defined as the ρ^* value between two configurations of a given automaton at *different* time steps, thereby computing the rate of cells that change or maintain their state at a given time.
2. *Inter-automata correlation*, henceforth inter-ρ^*, computed as the ρ^* value between configurations of two different automata at a given time step. The inter-automata correlation enables the pairwise comparison of the state of two automata and the quantification of the differences between them.

12.4 Effects of the Update Method on Synchronisation of Behaviour

In this section, we analyse the sensitivity of the cellular automata model to different update methods. We investigate: (i) if global synchronisation can arise from local pulse-coupling, (ii) the properties of behaviour under synchronous and asynchronous conditions, and (iii) the implications of using different alternatives to synchronous updating, including to what extent the emergence of synchronisation is affected by the way cells are updated. In addition, we analyse the interplay between different

configurations of parameters in the individual rules of the cells and the characteristics of the global behaviour.

12.4.1 Methods

The experiments are conducted using a fixed lattice with size $N = 10 \times 10$ and toroidal boundary conditions. We conduct experiments using the five different update methods described in Sect. 12.2.2, namely: (i) synchronous, (ii) α-asynchronous, (iii) κ-scaling, (iv) random order asynchronous, and (v) line-by-line sweep. For the α-asynchronous method and the κ-scaling method, the synchrony rate α is varied from $\alpha = 0.99$ (almost synchronous) to $\alpha = 1/N = 0.01$ (approximates sequential, fully asynchronous updating), in decrements of 0.01.

For each configuration in each set of experiments, we conduct 100 independent runs. Each automaton is initialised with a random configuration. A random activation level $\in [0, 1]$ sampled from a uniform distribution is independently assigned to each oscillator at the beginning of each run. The oscillation period T is set to 50 time steps. The coupling constant ϵ is set to 0.1. With respect to the oscillators' dynamics, as defined in Eq. 12.3, we use the linear dynamics given by the pulse-coupling function $g(x) = x$. We experiment with different coupling constants ϵ and distinct pulse-coupling functions in Sect. 12.4.2.2. Each run lasts 50,000 time steps.

12.4.2 Results

Table 12.1 lists the transient time necessary for the automata to synchronise when subject to different update methods. The synchronisation process is, on average, slightly faster if there is a fixed update order, either implicit as in the case of the synchronous update, or explicit as in the line-by-line sweep. Results show that a fixed update order is not always beneficial, as there is convergence in only 92 % of the runs conducted using each method. On the other hand, the random order method displays a convergence rate of 100 %. Differences in the convergence rate of the synchronous and the line-by-line sweep methods are statistically significant

Table 12.1 Transient time and convergence rate of systems under: (i) synchronous update, (ii) line-by-line sweep, and (iii) random order asynchronous update

Update Method	Average	Std. Dev.	Shortest	Longest	Convergence %
Synchronous	985	320	509	2,382	92
Line-by-line	897	292	380	1,793	92
Random Order	1,118	518	378	3,065	100

The values listed are the result of 100 independent runs for each experimental configuration

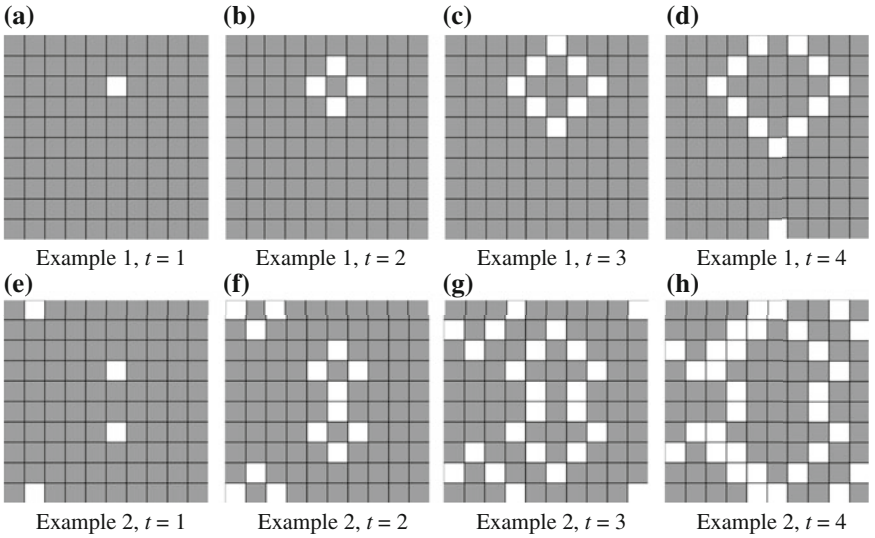

Fig. 12.4 Two examples of the field-based wave-like effect exhibited under synchronous updates. Cells in state 1 (resp. 0) are *coloured* in *white* (resp. *black*), a convention that is maintained throughout the study

with respect to the convergence rate of the random order method (two-tailed Fisher's exact test, $\rho = 6.8 \times 10^{-3}$).

The synchronous and line-by-line sweep methods lead to periodic vibration patterns that repeat every $T = 50$ time steps. Nonetheless, the spatiotemporal patterns exhibited by the automata are due to spurious effects caused by the update method. Two examples are shown in Fig. 12.4 for the synchronous case. The first example describes one particular run in which the automaton evolved into a behaviour with perfectly synchronised neighbour-to-neighbour influence. The first oscillator to fire is the origin of a field-based wave propagation phenomenon (see Sect. 12.2.1 for the definition of a *field*). When the wave starts, all oscillators have activation levels close to the firing threshold. After the firing of the first oscillator, each set of neighbours that have not yet fired will fire in consecutive time steps. The wave propagates throughout the system until all oscillators have fired. In this way, after the initial cell c becomes active at time step t, the behaviour of the automaton at time step $t + i$ will be a field $\varphi(c, i)$, $\forall i \leq 10$. If two or more oscillators fire in the initial time step, as shown in Fig. 12.4e, each of these activations starts a local field-based wave-like phenomenon.

The wave effect is due to the characteristics of the synchronous update method. The state of a given cell at time t can only be perceived by its neighbourhood at time $t + 1$, and therefore the activation of pulse-coupled oscillators is always taken into account with a latency of one time step by its neighbours. As a result, adjacent cells can only become active at consecutive time steps. Depending on how many cells

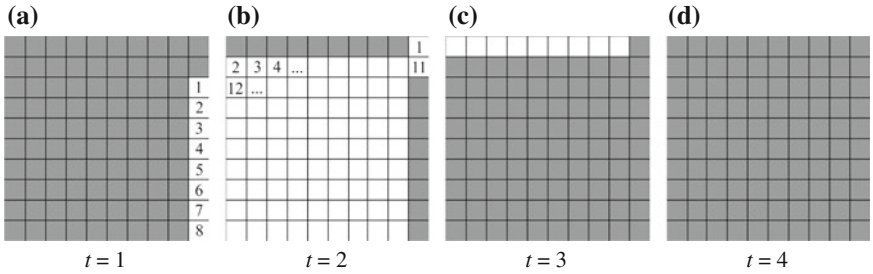

Fig. 12.5 Examples of the domino-like effect caused by the line-by-line sweep. In the first two time steps, $t = 1$ and $t = 2$, the order by which the cells become active in the corresponding time step is marked numerically. Cells in state 1 (resp. 0) are *coloured* in *white* (resp. *black*)

initiate the collective oscillation process and on how many neighbours each cell can cause to activate, the speed of collective oscillation after synchronisation may require from 6 time steps (Fig. 12.4, example 2) to 11 time steps (Fig. 12.4, example 1).

By analysing the sequence of activation of cells at a microscopic level, we observe that, under a line-by-line sweep, there is also a spurious correlation between the update order and the spatiotemporal patterns of the automata. The fixed sequential update order always causes a domino-like effect in the activations of the cells. Figure 12.5 exemplifies the domino-like effect during a collective oscillation behaviour. The order by which the cells become active in the first two time steps is marked numerically (Fig. 12.5a, b). Once a cell c is updated and becomes active, each neighbour of c will also become active when it is updated. In other words, the *directionality* of the influence between cells is solely due to the update order. Depending on the spatial position of the first cell to become active, the speed of collective oscillation varies from $N = 1$ time steps, if the cell is in the top-left corner of the automaton, to $N = 3$ time steps, which is the most frequent behaviour.

Increasing stochasticity in the update process through a continuously randomised order slightly delays convergence by augmenting the transient time. As listed in Table 12.1, random order asynchronous update yields an average transient time of 1,118 time steps. However, unlike the previously analysed update methods, the random order asynchronous scheme does not introduce any artefactual correlation in the dynamics of the cellular automata. Firstly, the automata always converge. Secondly, analysis of the spatiotemporal evolution of the automata shows that every 50 time steps, *all* cells become active in one time step, and therefore there is always a speed of collective oscillation $s = 1$. After the transient time $t_{transient}$ has elapsed, the asymptotic behaviour of synchronised automata is equal, which is detected by the stabilisation of the inter-ρ^* value $=1$ when comparing configurations at each time step $t_{transient} + i$, with $t_{transient} + i \leq 50{,}000$.

The collective oscillation under the random order method is illustrated in Fig. 12.6 through the intra-ρ^* values of one run. The transient time is of approximately 800 time steps. Variations of the intra-ρ^* value until time step 500 indicate that cells become active with no particular order or coupling. Afterwards, oscillators minimise phase

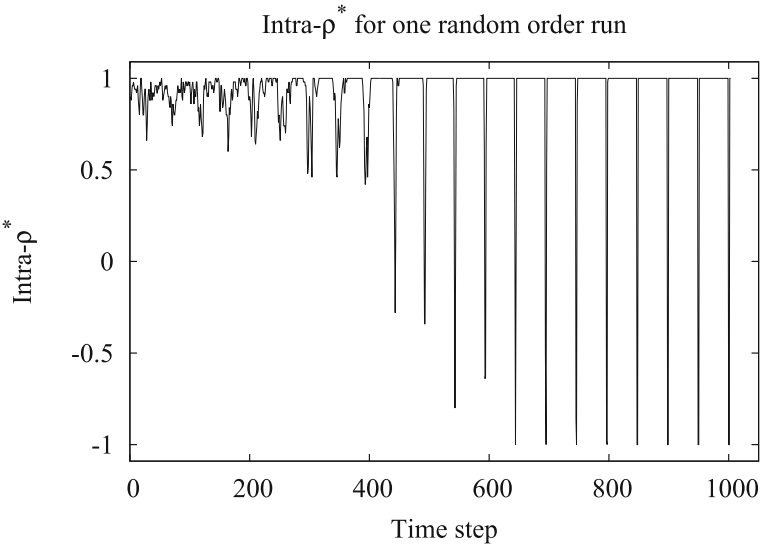

Fig. 12.6 Example of intra-CA correlation over the course of 1,000 time steps under the random order update method. The automata is synchronised after time step 800. Each peak of the intra-ρ^* value from 1 to -1 and vice-versa that occurs every 50 time steps represents one instance of collective oscillation (see text)

differences, and cells start to exhibit a progressively more synchronised behaviour. At time step 800, cells are completely synchronised based on an asynchronous process and local interactions. Each peak of the intra-ρ^* value from 1 to -1 and vice-versa represents one instance of collective oscillation. At this point, all cells are in state 0, i.e., no cell is active. In the following time step, all cells become active and therefore intra-ρ^* value $= -1$. A value of -1 is also measured in the subsequent time step because all cells move from an active to an inactive state, after which the intra-ρ^* value $=1$ is maintained until the next collective activation. The behaviour is then repeated every 50 time steps until the end of the simulation.

At a microscopic scale, during the activation time step the order by which the cells become active is determined by the randomly generated update sequences. These results support the work of Cornforth et al. [13] and Harvey and Bossomaier [48], among others, which have suggested that random order methods can generate quasi-cyclic behaviour. Under the light of the random order method, the automata can be said to display a truly emergent synchronised behaviour. One positive effect of this update method is that, as mentioned above, it removes the spurious effects that could be linked to a particular update order. Secondly, the random order scheme appears to be more faithful to the physical reality of the system being modelled than both the synchronous and the line-by-line methods. As initially argued by Kanada [12], the fact that the major source of randomness lies in the order of state transitions of the cells allows to take into account the microscopic structure of systems, and is therefore suitable for modelling purposes. However, it should be noted that in a

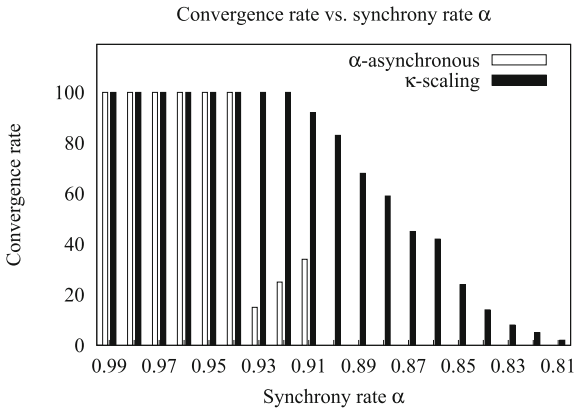

Fig. 12.7 Convergence rate of α-asynchronous and κ-scaling systems when subject to distinct degrees of asynchrony. Convergence cannot be ensured for α-asynchronous systems if $\alpha \leq 0.93$, and for κ-scaling systems if $\alpha \leq 0.91$

real distributed system, it may be difficult to ensure a completely random *sequential* updating order.

12.4.2.1 α-dependency

In this section, we analyse the behavioural response of the cellular automata when subject to α-asynchrony-based update methods. Results show that under the α-asynchronous and the κ-scaling methods, the behaviour of the automata significantly depends on the degree of asynchrony in their update. With increasing α-asynchrony, the automata evolve a behaviour noticeably different from that produced by the three update methods analysed above. As shown in Fig. 12.7, for $0.90 < \alpha \leq 0.93$, the convergence rate of α-asynchronous systems varies from 15 % to 34 %, which is significantly lower than the convergence rate of automata subject to higher α values (two-tailed Fisher's exact test, $\rho < 1.0 \text{ x } 10^{-4}$ for all $0.90 < \alpha \leq 0.93$). In effect, for $\alpha \leq 0.90$, α-asynchronous systems can no longer synchronise in the 50,000 time steps limit. Performing a temporal compensation and corresponding normalisation of the sampling differences by means of the κ-scaling method counteracts to some extent the effects of α-asynchrony. The automata can always overcome asynchrony for $\alpha > 0.91$. As the synchrony rate is lowered, the convergence rate of κ-scaling systems decreases proportionally to α.

From a practical point of view, α-asynchronous and κ-scaling methods show how sensitive cellular automata can be to asynchrony. Table 12.2 lists the average transient time and the average period for $0.99 \leq \alpha \leq 0.94$. Aside from the aforementioned convergence rate, asynchrony by means of less cell activity significantly increases both the transient time and the average period length between consecutive group-level activations. Even for minimal asynchrony, i.e., $\alpha = 0.99$, the average period is 128 time steps for α-asynchronous systems and 124 time steps for κ-scaling systems. With increasing asynchrony, the automata become increasingly lagged thus resulting

Table 12.2 Transient time and length of the period between consecutive group-level oscillations

Synchrony rate α	α-asynchronous		κ-scaling	
	Transient time	Period	Transient time	Period
0.99	1,090	128	1,152	124
0.98	1,437	345	1,340	302
0.97	1,989	863	1,855	680
0.96	3,495	2,294	2,404	1,471
0.95	12,189	6,519	3,876	2,755
0.94	19,278	9,753	4,931	4,387

Values are the average of 100 independent runs for each experimental configuration

in significantly higher transient times and periods between collective oscillations. For $\alpha = 0.94$, α-asynchronous systems have a transient time of approximately 19,278 time steps and a period of 9,753 time steps. κ-scaling systems are less affected as they display a transient time of 4,931 time steps and a period of 4,387 time steps.

Our analysis shows that: (i) the assessed α-asynchrony-based update methods do not enable the systems to evolve towards a global vibration pattern, and (ii) increasing asynchrony further degrades behaviour. In fact, the existence of contrasting behaviours under α-asynchronous dynamics, even when using moderate asynchrony values, indicates a hidden sensitivity that may need to be handled carefully depending on the context.

Overall, the results presented in this section corroborate that not only is asynchrony relevant, but that the *type* of asynchrony can have significant effects when modelling multiagent or multirobot systems by means of discrete dynamical systems. This question takes a special importance due to the fact that, as argued by Cornforth et al. [13], multiagent systems are usually modelled based on the synchronous update method. This begs the question: when modelling multiagent systems that have to achieve synchronisation or consensus, in which cases is the multiagent system robust to modifications in the updating? Complementarily, in such multiagent systems, to what extent is the behaviour exhibited under asynchronous dynamics dependent on the properties of the local transition function? The following section addresses the second question by analysing the robustness and the global dynamics of the system when subject to variations in the main parameters of the pulse-coupled oscillators: the coupling constant ϵ and the coupling function $g(x)$.

12.4.2.2 Assessing the Dynamics of the Random Order Method

The behaviour of the oscillators as described in Eq. 12.3 depends on the interactions between neighbouring cells. The strength of each interaction is controlled by the coupling constant ϵ and by the coupling function $g(x)$. In this section, we assess in which conditions the variation of ϵ and of $g(x)$ alters the dynamics of the automata.

Table 12.3 Convergence rate of different pulse-coupling dynamics as a function of the coupling constant ϵ

Coupling constant ϵ	Convergence rate (%)		
	Linear function	Sigmoid function	Sine function
0.01	14	0	0
0.02	99	33	0
0.05	100	100	0
0.1	100	100	0
0.2	100	100	0
0.5	100	100	100
1.0	100	100	100

For each experimental configuration, the convergence rate is measured by performing 100 independent runs

Our goal is to examine the robustness or sensitivity of the update method to changes in the rules that govern the behaviour of cells.

We conduct the experiments using the update method that yielded the most reliable result, namely the random order asynchronous method. The effects of the coupling constant are assessed for $\epsilon \in \{0.01, 0.02, 0.05, 0.10, 0.20, 0.50, 1.0\}$. We also study three pulse-coupling functions. The choice of the coupling function was made to test different dynamics, namely: (i) linear dynamics with $g(x) = x$, (ii) even symmetrical dynamics with respect to $x = 0.5$ based on a sine function $g(x) = \sin(\pi x)$, and (iii) "S-shaped" dynamics given by the sigmoid function $g(x) = \frac{1}{1+e^{(6-12x)}}$. The three functions are shown in Fig. 12.8. For each experimental configuration, we perform 100 independent runs. Oscillators are initialised as defined in Sect. 12.4.1, i.e., with random initial activation levels, and with the period T set to 50 time steps. Each run lasts 50,000 time steps.

Table 12.3 summarises the convergence rate when ϵ and $g(x)$ are varied. The linear pulse-coupling is the most robust function as it shows the highest overall convergence rate in our experiments. Except for the lowest value $\epsilon = 0.01$, convergence is almost always ensured. The sigmoid function and the sine function ensure con-

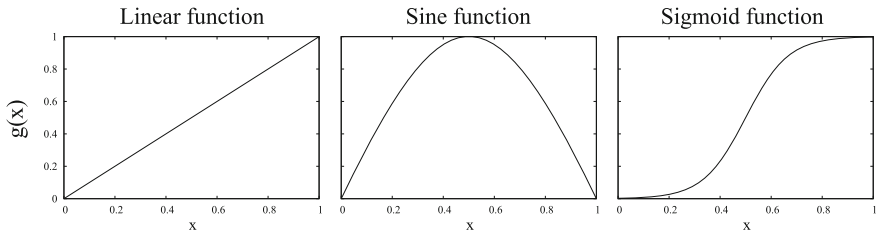

Fig. 12.8 The three models of pulse-coupling functions assessed. From *left* to *right*: linear dynamics, even symmetrical dynamics with respect to $x = 0.5$ based on a sine function $g(x) = \sin(\pi x)$, and "S-shaped" dynamics given by the sigmoid function $g(x) = \frac{1}{1+e^{(6-12x)}}$

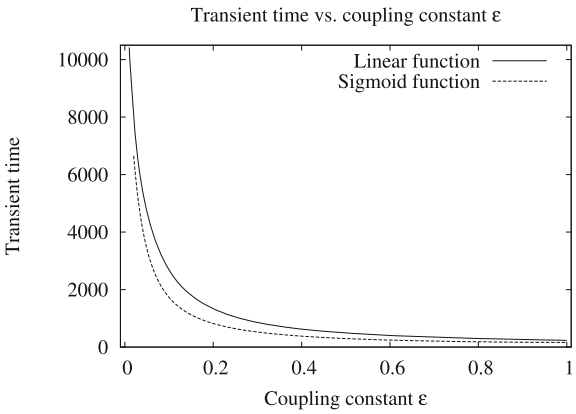

Fig. 12.9 Average transient time as a function of the coupling constant ϵ. For the linear function, the average transient time varies from 10,423 time steps for $\epsilon = 0.01$ to 234 time steps for $\epsilon = 1$. For the sigmoid function, the average transient time is of 6,651 time steps for $\epsilon = 0.02$ and of 161 time steps for $\epsilon = 1.0$. For the sine function (not plotted), the average transient time varies from 326 time steps ($\epsilon = 0.5$) to 96 time steps ($\epsilon = 1$)

vergence if $\epsilon \geq 0.05$ and $\epsilon \geq 0.5$, respectively. With respect to the time necessary for convergence, Fig. 12.9 shows the average transient time for the linear function and for the logistic function when using different coupling values. For these two functions, the lowest transient time is given when the oscillators are strongly coupled. For $\epsilon = 1$, the average transient time is 234 time steps for the linear function and 161 time steps for the sigmoid function. The results suggest that high values of ϵ are preferable in the synthesis of synchronised behaviour. For the two functions, the transient time decreases exponentially with the increase of the coupling constant until $\epsilon = 0.6$, after which point the values vary significantly less.

Despite the substantial reduction of the transient time due to the increase of the coupling constant when using either a linear function or a sigmoid function, the fastest overall convergence is given by the even symmetrical dynamics of the sine function. The average transient time varies from 326 time steps for $\epsilon = 0.5$ to 96 time steps for $\epsilon = 1$. The reduction in transient time is due to the characteristics of the sine function, which enable a more effective adjustment of the activation level. The line of symmetry dictates that increasingly higher adjustments are made to the activation level for $0 < x < 0.5$. For $x > 0.5$, the closer the activation level x is of the threshold of 1, the more subtle the adjustment is.

The analysis conducted in this section shows that the parameters ϵ and $g(x)$ can have a profound effect on the convergence rate and transient time of the automata. We estimated and analysed the spectrum of convergence rates and transient times. Firstly, the results show that increasing the strength of the interaction between neighbouring cells is beneficial as if ϵ is large, the oscillators tend to synchronise faster. Secondly, coupling functions with different properties enable slower or faster convergence. Thirdly, it should also be noted how low coupling values affect the behaviour exhib-

ited by an automaton. For $\epsilon = 0.01$ and $\epsilon = 0.02$, the speed of collective oscillation varies from 1 to 11 time steps for the linear function and from 1 to 7 time steps for the sigmoid function. In other words, low coupling constants cause instabilities in the systems as the automata evolve towards different configurations. Consequently, the periodic, quasi-cyclic patterns and the repetition of series of configurations are altered by using different parameters.

12.4.3 Summary

In this section, we examined the effects of the update method in the synthesis of synchronised behaviour. We showed that in the modelling of multirobot or multiagent systems, not only is asynchrony relevant, but that the type of asynchrony has profound effects on dynamics. These results are especially important if we consider that multiagent systems are usually modelled using synchronous update methods [13].

In our experimental setup, the random order asynchronous method was shown to be the most reliable method. We showed that both the synchronous and the line-by-line schemes introduce artificial structure in the evolution of our cellular automata model. We also showed that, in our modelling scenario, α-asynchronous and κ-scaling methods can also have significant effects on the behaviour of the system. The two methods exhibited a hidden sensitivity, even for high asynchrony values, that should be handled carefully when modelling systems that are continuous in time and space.

To conclude the section, we experimentally analysed the importance of the coupling constant and of the pulse-coupling function. Chiefly, our results showed that: (i) stronger interactions between neighbours are preferable, as they significantly increase both the convergence rate and the transient time, and (ii) that the type of dynamics of the coupling function play a key role in the synchronisation process.

12.5 Perturbing the Topology

In this section, we investigate how topology perturbations modify the evolution of synchronised behaviour in our cellular automata model. Our goal is to determine what happens when the cellular automata are no longer perfectly synchronous and the topology is perturbed, i.e., to estimate the stability or sensitivity of the systems when topology characteristics become irregular.

12.5.1 Methods

We perturb the topology by definitively removing connections between cells. Let $G = (V, E)$ be the oriented graph that represents the connections between

cells. For all cells $c_i, c_j \in V$, with $c_i \neq c_j$, the connection $(c_i, c_j) \in E$ if and only if c_j belongs to the neighbourhood of c_i. The oriented graph with perturbed topology $G_{pert} = (V, E_{pert})$ is obtained by assigning to each cell $c_i \in V$ an independent probability of removing a randomly chosen connection between c_j and one of its neighbours c_j. The parameter p_{cr} is defined as the *connection removal* rate. It should be noted that the discrete pulse-coupled model defined in Eq. 12.3 is a totalistic rule in the sense that it takes into account the total of active neighbours. A neighbour that is not sensed due to a connection removed is considered as being always in state 0. In this way, perturbing the topology of the automata is conceptually similar to simulating faults in the sensors of the robots, one of the central issues with autonomous robots [49].

We conduct two series of experiments, henceforth *dynamic* perturbations setup and *fixed* perturbations setup. In the dynamic perturbations setup, there is a continual removal of connections at *each* time step. Our objective is to investigate the behaviour of the cellular automata when topology characteristics are continuously modified throughout time. It should be noted that in this experimental setup, the connections graph may become disconnected as a result of the deletion of connections between cells. Therefore, the major requirement in the task is for each cell to synchronise with its neighbours before the neighbours stop being sensed due to the lack of connections. On the other hand, in the fixed perturbations setup we analyse the convergence properties of the systems when the topologies are made irregular *but* do not change during the course of a simulation. In this setup, connections are probabilistically removed according to the connection removal rate p_{cr} before the simulation starts.

The experimental protocol is here defined using a lattice with size $N = 10 \times 10$ and toroidal boundary conditions. Experiments are conducted using the random order asynchronous method and the linear pulse-coupling function $g(x) = x$, as this was shown to be the most robust configuration. In the dynamic perturbations setup, the connection removal rate p_{cr} is varied in $[0.0, 0.1]$ in steps of 0.002. As the convergence properties of the system is dependent on the strength of the interactions between neighbouring cells, we also vary the coupling constant $\epsilon \in [0.1, 1.0]$ in steps of 0.1. In the fixed perturbations setup, the connection removal rate is larger because connections are only removed once, i.e., before the simulation starts. The connection removal rate is varied in $[0.0, 0.75]$ in steps of 0.05. We experimentally verified that for $p_{cr} > 0.75$, the graph tends to be disconnected, which we do not allow because convergence would not be possible. For each configuration in each set of experiments, we conduct 100 independent runs. The parameters and initialisation of the automata follow the experimental setup described in Sect. 12.4.1. Each run lasts 50,000 time steps.

12.5.2 Results

We start by analysing the results of the dynamic perturbations setup. In Fig. 12.10, we separately represent the convergence rate and the transient time as a function of

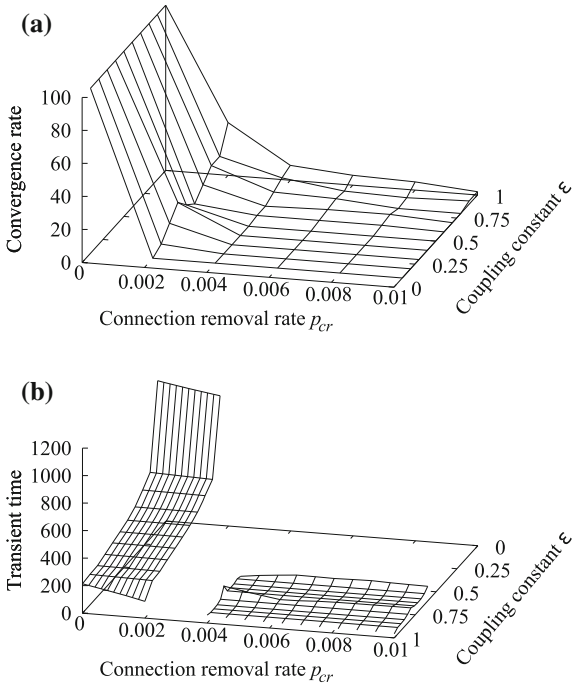

Fig. 12.10 Dynamic perturbations setup. Sampling surface of: (*a*) the convergence rate sampling surface for the dynamic perturbations setup, and (*b*) the average transient time sampling surface for the dynamic perturbations setup. The values displayed in each sampling surface are a function of both the coupling constant ϵ and the connection removal rate p_{cr}. **a** for $p_{cr} > 0.01$ the sampling surface is flat, which indicates 0 % of convergence—not shown for better visualisation of the sampling surface. **b** the sampling surface shows a discontinuity of behaviour for $0.002 < p_{cr} < 0.004$, which is marked by separating the sampling surface into two parts. Note that the range of the y-axis is reversed, i.e., values regarding the coupling constant ϵ are plotted from $y = 1$ to $y = 0$

the connection removal rate p_{cr} and of the coupling constant ϵ. Each set of values obtained is represented in a three-dimensional space, which is projected on a two-dimensional *sampling surface*.

Results show that the cellular automata are sensitive to perturbations in the topology. In general, higher coupling values ϵ enable the automata to synchronise more often. However, even for the smaller value of $p_{cr} = 0.002$ considered, the maximum convergence rate is of 32 % (for $\epsilon = 1.0$). For $p_{cr} = 0.004$, there is no synchronisation of behaviour for $\epsilon \leq 0.5$ and the highest convergence rate is of 9 % ($\epsilon = 1.0$). For higher connection removal rates, the convergence rate continuously decreases. For $p_{cr} > 0.01$, there is no convergence and the sampling surface is flat and horizontal.

The impact of removing connections is two-fold. An interesting effect of perturbing the topology is that, as shown in Fig. 12.10b, increasing rates of connection removal can harm the overall convergence rate *but* reduce the transient time when the automata do converge. Visual examination of the transient time sampling surface

shows a discontinuity of behaviour, which is marked by separating the surface into two parts. If no connections are removed, the minimum average transient time is 213 time steps ($\epsilon = 1$). For $p_{cr} = 0.002$, the transient time evolves in a similar manner except that the minimum average transient time is of 122 time steps, again for $\epsilon = 1.0$. For the critical values of $p_{cr} \geq 0.004$, for which there is no convergence for $\epsilon \leq 0.5$, the transient time is consistently low and decreases to an average of 30 time steps for the case $p_{cr} = 0.01$, $\epsilon = 1.0$.

The results obtained in the dynamic perturbations setup are the consequence of multiple intricate factors related to local synchronisation of cells. Removing connections can either deteriorate or accelerate convergence depending on the context of the cells. For example, consider a cell c_i and two of its neighbours, c_j and c_k. Cells c_i and c_j have a small lag between their activations, i.e., they are not entirely synchronised, and c_k is significantly delayed with respect to the firings of c_i and c_j. If the connection (c_i, c_j) is removed, c_i will have to synchronise its behaviour *only* with c_k, which will require an amount of time that depends on the phase differences between the two cells. During this time, depending on the connection removal rate, the connection (c_i, c_k) may be removed before the cells are synchronised, causing them to remain unsynchronised throughout time. As a result, there will be no convergence in the automaton.

If the first connection removed refers to two neighbours that have a large phase difference, namely c_i and c_k, synchronisation is accelerated because c_i synchronises with c_j without the "interference" of c_k. If the automata at a global level are subject to more acceleration than deterioration phenomena, then a self-organised behaviour will emerge faster as the result of the interaction between the cells at a local level. On the other hand, if there are more deterioration than acceleration phenomena in the relation between the cells, convergence is made progressively more difficult because the graph becomes increasingly disrupted and potentially disconnected. Therefore, in the dynamic perturbations setup, the acceleration and deterioration aspects described above are the result of a *continuous* removal of connections. Nonetheless, one important question remains: if the topology is perturbed but afterwards remain unchanged, under which conditions can the cells synchronise their behaviour?

In the fixed perturbations setup, the automata *always* converge regardless of the experimental configuration, which results in a completely horizontal sampling surface. For the most extreme case assessed, $p_{cr} = 0.75$, on average 75 out of 400 connections are removed from the initial topology, equivalent to 18.75 %. This aspect is indicative of the robustness of the cellular automata model and of its ability converge, as long as the connection graph is connected.

Figure 12.11 shows the sampling surface describing the evolution of the transient time. In the fixed perturbations setup, contrary to the dynamic perturbations experiments, the transient time tends to increase if higher ϵ and p_{cr} values are used. This increase is more accentuated for coupling constants around $\epsilon \approx 0.75$. By analysing the speed of collective oscillation and the period between consecutive oscillations, we observe that for $p_{cr} > 0.4$ and $\epsilon > 0.75$, the system becomes unstable. For these values, the average period varies from 100 to 180 time steps, and collective oscillation can require up to 10 time steps. The results show the interplay between the

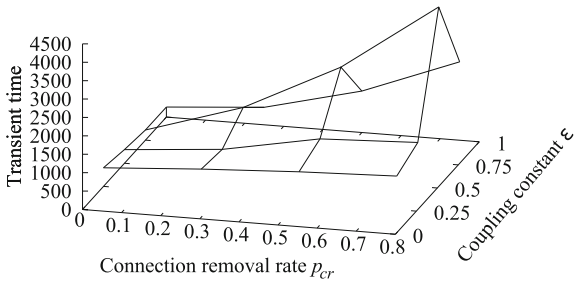

Fig. 12.11 Fixed perturbations setup. Sampling surface of the average transient time. The values displayed in the sampling surface are a function of both the coupling constant ϵ and of the connection removal rate p_{cr}

coupling constant and the *degree* of connectivity of the connections graph, thereby indicating that the topology indeed plays a central role in the synchronisation properties of the cellular automata. In automata with regular topologies such as those used in Sect. 12.4, higher coupling constants ϵ are beneficial and accelerate convergence. However, if the connectivity of the automata is irregular, then setting ϵ too high is problematic because each cell has a significant effect on its neighbours. The high degree of influence in the behaviour of neighbouring cells leads to phase instability, and can drive the system towards either acceleration of convergence or deterioration of behaviour.

12.5.3 Summary

In this section, we analysed the effects of perturbing the cellular automata topology by probabilistically removing connections between neighbouring cells. Based on extensive numerical simulations, we showed that topology characteristics are important for the emergence of synchronised behaviour. If the irregularity of the topology increases over time, the impact is often two-fold. The convergence rate decreases with the increase of irregularity but the transient time is effectively smaller when the automata manage to converge. On the other hand, if the degree of irregularity is kept fixed, convergence is ensured as long as the connections graph does not become disconnected. Additionally, results showed that the strength of the interaction between neighbouring cells by means of the coupling constant ϵ plays a central role in the two circumstances. In particular cases, high values of ϵ either accelerate synchronisation of behaviour, or introduce instabilities in the system. These fluctuations significantly increase the time necessary for convergence and the way by which instances of collective behaviour are produced. Therefore, depending on the topology of the cellular automata, it is necessary to compromise between the strength of local interactions and the degree of connectivity of the network.

12.6 Discussion

A long-standing question in the field of complex systems is determining whether cellular automata are appropriate modelling tools for multiagent systems or if they are not sufficiently expressive. To answer the question, it is first necessary to carefully assess the robustness of cellular automata in a modelling context, and to cover a broad number of conditions in order to identify the *limits* of the modelling tool.

This chapter is a contribution towards the understanding of the effects of the update method and of the topological structure when modelling real-world complex systems with cellular automata. We analysed the collective dynamics of a group of stationary robots inspired by studies in mixed natural-artificial societies [6]. The system was composed by 100 robots, and the behaviour of individual robots was modelled as a pulse-coupled oscillator. We addressed the problem of self-organised synchronisation, in which robots had to adjust their behaviour to produce a population-wide common vibration pattern based on local interactions. We focused on two fundamental aspects: (i) the effects of different update methods, including the interplay between parameters of local rules and the global behaviour, and (ii) the transition from regular to irregular grids by means of dynamic perturbations and fixed perturbations.

The first set of experiments outlined in this chapter demonstrated the impact of five different update methods. Results showed that the way robots synchronise their behaviour can be bounded by the singular properties of the update method. Modifying the state of the cells according to a random update method was shown to be the most robust approach with respect to producing the desired dynamics in the multirobot system. In the second part of the chapter, we conducted a systematic study on the robustness of cellular automata to topology perturbations under a number of different conditions. We concluded that the topology characteristics have various effects on the behaviour of the automata. When subject to topology perturbations, behaviour exhibited may vary from extremely sensitive to extremely robust. Therefore, although few researchers have studied this aspect [29], the topology structure is deemed essential for future studies in modelling the behaviour of multiagent systems with cellular automata.

The broader agenda for future study on more realistic modelling of real-world complex systems with cellular automata is to gain new insights on the emergent behaviour of large-scale multirobot systems, both stationary and mobile, by analysing them from a dynamical systems perspective. In this respect, the topology perturbation experiments can be extended to simulate scenarios in which a failed robot is repaired or replaced by a new one, or the fault is transient, and therefore the cellular automata recover part of their previously lost connections. We also intend to address increasingly more complex tasks in which robots have to achieve synchronisation or consensus. Instead of manually tuning the parameters of individual behaviour, our goal is to synthesise by means of machine learning techniques, the specific parameters of the pulse-coupled oscillators (or other models) that can generate a set of target spatiotemporal dynamics. One important part of our future work is to analyse

the generality and robustness of the learned parameters by assessing them under different update methods and topological structures.

Acknowledgments This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT) under the grants SFRH/BD/89573/2012, PEst-OE/EEI/UI0434/2011, PEst-OE/EEI/LA0008/2013, and EXPL/EEI-AUT/0329/2013, and by the European Union —Information and Communication Technologies project 'ASSISIbf', no. 601074.

References

1. Fatès, N., Chevrier, V.: How important are updating schemes in multi-agent systems? An illustration on a multi-turmite model. In: 9th International Conference on Autonomous Agents and Multiagent Systems, pp. 533–540. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2010)
2. Spicher, A., Fatès, N., Simonin, O.: Translating discrete multi-agents systems into cellular automata: application to diffusion-limited aggregation. In: Filipe, J., Fred, A., Sharp, B. (eds.) Agents and Artificial Intelligence. Communications in Computer and Information Science, vol. 67, pp. 270–282. Springer, Berlin, Germany (2010)
3. Tosic, P.: On modeling large-scale multi-agent systems with parallel, sequential and genuinely asynchronous cellular automata. Acta Phys. Pol. B Proc. Suppl. **4**(2), 217–235 (2011)
4. Bouré, O., Fatès, N., Chevrier, V.: First steps on asynchronous lattice-gas models with an application to a swarming rule. Nat. Comput. **12**(4), 551–560 (2013)
5. Chevrier, V., Fatès, N.: Multi-agent systems as discrete dynamical systems: Influences and reactions as a modelling principle. Technical report, INRIA-LORIA (2008)
6. Schmickl, T., Bogdan, S., Correia, L., Kernbach, S., Mondada, F., Bodi, M., Gribovskiy, A., Hahshold, S., Miklic, D., Szopek, M., Thenius, R., Halloy, J.: ASSISI: Mixing animals with robots in a hybrid society. In: Second International Conference on Biomimetic and Biohybrid Systems. volume 8064 of Lecture Notes in Computer Science, pp. 441–443. Springer, Berlin, Germany (2013)
7. Pikovsky, A., Rosenblum, M., Kurths, J.: Synchronization: A Universal Concept in Nonlinear Sciences. Cambridge University Press, Cambridge, UK (2003)
8. Tyrrell, A., Auer, G.: Imposing a reference timing onto firefly synchronization in wireless networks. In: 65th IEEE Vehicular Technology Conference, pp. 222–226. IEEE Press, Piscataway, NJ (2007)
9. Werner-Allen, G., Tewari, G., Patel, A., Welsh, M., Nagpal, R.: Firefly-inspired sensor network synchronicity with realistic radio effects. In: 3rd International Conference on Embedded Networked Sensor Systems, pp. 142–153. ACM Press, New York, NY (2005)
10. Christensen, A.L., O'Grady, R., Dorigo, M.: From fireflies to fault-tolerant swarms of robots. IEEE Trans. Evol. Comput. **13**(4), 754–766 (2009)
11. Mirollo, R., Strogatz, S.: Synchronization of pulse-coupled biological oscillators. SIAM J. Appl. Math. **50**(6), 1645–1662 (1990)
12. Kanada, Y.: The effects of randomness in asynchronous 1D cellular automata. Technical report, Tsukuba Research Center (1997)
13. Cornforth, D., Green, D.G., Newth, D.: Ordered asynchronous processes in multi-agent systems. Phys. D Nonlinear Phenom. **204**(1), 70–82 (2005)
14. Correia, L.: Self-organised systems: fundamental properties. Revista de Ciências da Computação **1**(1), 9–26 (2006)
15. Fatès, N.: A guided tour of asynchronous cellular automata. In: 19th International Workshop on Cellular Automata and Discrete Complex Systems. volume 8155 of Lecture Notes in Computer Science, pp. 15–30. Springer, Berlin, Germany (2013)

16. Bandini, S., Bonomi, A., Vizzari, G.: An analysis of different types and effects of asynchronicity in cellular automata update schemes. Nat. Comput. **11**(2), 277–287 (2012)
17. Dennunzio, A., Formenti, E., Manzoni, L.: Computing issues of asynchronous CA. Fundam. Informaticae **120**(2), 165–180 (2012)
18. Dennunzio, A., Formenti, E., Manzoni, L., Mauri, G.: m-asynchronous cellular automata: from fairness to quasi-fairness. Nat. Comput. **12**(4), 561–572 (2013)
19. Fatès, N., Morvan, M.: An experimental study of robustness to asynchronism for elementary cellular automata. Complex Syst. **16**(1), 1–27 (2005)
20. Ingerson, T.E., Buvel, R.L.: Structure in asynchronous cellular automata. Phys. D Nonlinear Phenom. **10**(1–2), 59–68 (1984)
21. Schönfisch, B., de Roos, A.: Synchronous and asynchronous updating in cellular automata. Biosystems **51**(3), 123–143 (1999)
22. Silva, F., Correia, L.: An experimental study of noise and asynchrony in elementary cellular automata with sampling compensation. Nat. Comput. **12**(4), 573–588 (2013)
23. Bersini, H., Detours, V.: Asynchrony induces stability in cellular automata based models. In: 4th International Conference on Simulation and Synthesis of Living Systems, pp. 382–387. MIT Press, Cambridge, MA, (1994)
24. Ruxton, G., Saravia, L.: The need for biological realism in the updating of cellular automata models. Ecol. Modell. **107**(2–3), 105–112 (1998)
25. Fatès, N.: Directed percolation phenomena in asynchronous elementary cellular automata. In: 7th International Conference on Cellular Automata for Research and Industry. volume 4173 of Lecture Notes in Computer Science, pp. 667–675. Springer, Berlin, Germany (2006)
26. Fatès, N.: Asynchrony induces second order phase transitions in elementary cellular automata. J. Cell. Automata **4**(1), 21–38 (2009)
27. Fatès, N., Regnault, D., Schabanel, N., Thierry, E.: Asynchronous behavior of double-quiescent elementary cellular automata. In: 7th Latin American Symposium. volume 3887 of Lecture Notes in Computer Science, pp. 455–466. Springer, Berlin, Germany (2006)
28. Regnault, D.: Proof of a phase transition in probabilistic cellular automata. 17th: International Conference on Developments in Language Theory. volume 7907 of Lecture Notes in Computer Science, pp. 433–444. Springer, Berlin, Germany (2013)
29. Fatès, N.: Critical phenomena in cellular automata: perturbing the update, the transitions, the topology. Acta Phys. Pol. B Proc. Suppl. **3**(2), 315–325 (2010)
30. Fatès, N.: Does life resist asynchrony? In: Adamatzky, A. (ed.) Game of Life Cellular Automata, chapter 14, pp. 257–274. Springer, London, UK (2010)
31. Fatès, N., Gerin, L.: Examples of fast and slow convergence of 2D asynchronous cellular systems. J. Cell. Automata **4**(4), 323–337 (2009)
32. Belgacem, S., Fatès, N.: Robustness of multi-agent models: the example of collaboration between turmites with synchronous and asynchronous updating. Complex Syst. **21**(3), 165–182 (2012)
33. Kitagawa, T.: Cell space approaches in biomathematics. Math. Biosci. **19**(1), 27–71 (1974)
34. Seybold, P., Kier, L., Cheng, C-K.: Simulation of first-order chemical kinetics using cellular automata. J. Chem. Inf. Comput. Sci. **37**(2), 386–391 (1997)
35. Rajewsky, N., Santen, L., Schadschneider, A., Schreckenberg, M.: The asymmetric exclusion process: comparison of update procedures. J. Stat. Phys. **92**(1–2), 151–194 (1998)
36. Strogatz, S., Stewart, I.: Coupled oscillators and biological synchronization. Sci. Am. **269**(6), 102–109 (1993)
37. Smith, H.M.: Synchronous flashing of fireflies. Science **82**(2120), 151–151 (1935)
38. Glass, L.: Synchronization and rhythmic processes in physiology. Nature **410**(6825), 277–284 (2001)
39. Strogatz, S.: From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators. Phys. D Nonlinear Phenom. **143**(1–4), 1–20 (2000)
40. Peskin, C.S.: Mathematical Aspects of Heart Physiology. Courant Institute Lecture Notes. Courant Institute of Mathematical Sciences, New York, NY (1975)

41. Bottani, S.: Pulse-coupled relaxation oscillators: from biological synchronization to self-organized criticality. Phys. Rev. Lett. **74**(21), 4189–4192 (1995)
42. Lucarelli, D., Wang, I-J.: Decentralized synchronization protocols with nearest neighbor communication. In: 2nd International Conference on Embedded Networked Sensor Systems, pp. 62–68. ACM Press, New York, NY, (2004)
43. Izhikevich, E.: Weakly pulse-coupled oscillators, FM interactions, synchronization, and oscillatory associative memory. IEEE Trans. Neural Networks **10**(3), 508–526 (1999)
44. Fuks, H., Skelton, A.: Orbits of the Bernoulli measure in single-transition asynchronous cellular automata. In: 17th International Workshop on Cellular Automata and Discrete Complex Systems, Discrete Mathematics and Theoretical Computer Science, pp. 95–112, (2011). Available from: http://www.dmtcs.org/dmtcs-ojs/index.php/proceedings/article/view/dmAP0107
45. Bouré, O., Fatès, N., Chevrier, V.: Probing robustness of cellular automata through variations of asynchronous updating. Nat. Comput. **11**(4), 553–564 (2012)
46. Hamming, R.W.: Error detecting and error correcting codes. Bell Syst. Tech. J. **29**(2), 147–160 (1950)
47. Silva, F., Correia, L.: A study of stochastic noise and asynchronism in elementary cellular automata. In: 10th International Conference on Cellular Automata for Research and Industry. volume 7495 of Lecture Notes in Computer Science, pp. 679–688. Springer, Berlin, Germany (2012)
48. Harvey, I., Bossomaier, T.: Time out of joint: attractors in asynchronous random boolean networks. In: 4th European Conference on Artificial Life, pp. 67–75. MIT Press, Cambridge, MA, (1997)
49. Carlson, J., Murphy, R.R., Nelson, A.: Follow-up analysis of mobile robot failures. In: IEEE International Conference on Robotics and Automation, pp. 4987–4994. IEEE Computer Society Press, Los Alamitos, CA, (2004)