Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 236285, 19 pages
http://dx.doi.org/10.1155/2015/236285

Research Article

Stabilization Methods for a Multiagent System with Complex Behaviours

Florin Leon

Department of Computer Science and Engineering, "Gheorghe Asachi" Technical University of Iași, D. Mangeron 27 Street, 700050 Iași, Romania

Correspondence should be addressed to Florin Leon; fleon@cs.tuiasi.ro

Received 29 November 2014; Accepted 27 April 2015

Academic Editor: Reinoud Maex

The main focus of the paper is the stability analysis of a class of multiagent systems based on an interaction protocol which can generate different types of overall behaviours, from asymptotically stable to chaotic. We present several interpretations of stability and suggest two methods to assess the stability of the system, based on the internal models of the agents and on the external, observed behaviour. Since it is very difficult to predict a priori whether a system will be stable or unstable, we propose three heuristic methods that can be used to stabilize such a system during its execution, with minimal changes to its state.

1. Introduction

Multiagent systems have enjoyed increasing popularity by both researchers and practitioners, due to the continuation of trends in computing such as ubiquity, interconnection, intelligence, delegation, and human-orientation [1]. Agent-based systems can be used as a tool to understand social and economic systems, since the aggregation of simple rules of behaviour for individual agents often leads to an extremely rich range of emergent behaviours, which can help to understand complex phenomena in human societies [2–12].

The actions of individual agents are often based on a certain process of decision making. Therefore, stability, intuitively understood as the property of a system to exhibit bounded behaviour [13], is one of the most desired features in multiagent systems, because of the importance of predicting their response to various conditions in the environment.

The initial motivation of developing the interaction protocol which will be analysed in the present paper was to design a set of simple interaction rules which in turn can generate, through a cascade effect, different types of overall behaviours, that is, stable, periodical, quasi-periodical, chaotic, and nondeterministically unstable. They can be considered metaphors for the different kinds of everyday social or economic interactions, whose effects are sometimes entirely predictable and can lead to an equilibrium while, in some other times, fluctuations can widely affect the system state, and even if the system appears to be stable for long periods of time, sudden changes can occur unpredictably because of subtle changes in the internal state of the system.

We organize our paper as follows. In Section 2, we review some of the related work in the areas of dynamical and evolutionary behaviours, stabilization, and phase transitions in multiagent systems. In Section 3, we describe the interaction protocol employed by the multiagent system under study. In Section 4, we present different interpretations of stability and suggest two methods to assess the stability of the system, based on the internal model and the external, observed behaviour. In Section 5, we give some empirical evidence about the difficulty of predicting a priori whether the system will be stable or unstable and present experiments that can help to identify the nature of phase transitions. Section 6 describes three heuristic methods for the stabilization of the system with minimal changes, along with case studies and discussion. Section 7 contains the conclusions of the paper and addresses some future lines of investigation.

2. Related Work

An important aspect of dynamical systems is the issue of stability. In the multiagent systems literature, the study of

stability is mainly applied for three broad types of problems. The first is the question of stability in evolutionary systems. The second is formation control, including the stability of simulated swarms of particles or groups of mobile robots. The third is the general issue of convergence of agent interactions, consensus in networks of agents modelled using graphs, and so forth.

Nonlinear effects are characteristic of evolutionary game theory [14], which aims to enhance the concepts of classical game theory [15] with evolutionary issues, such as the possibility to adapt and learn. In general, the fitness of a certain phenotype is, in some way, proportional to its diffusion in the population. The strategies of classical game theory are replaced by genetic or cultural traits, which are inherited, possibly with mutations. The payoff of a game is interpreted as the fitness of the agents involved [16].

Many such models have been proposed, based on the different ways in which agents change their behaviours over time. Among them we can mention replicator dynamics [17, 18], its replicator-mutator generalization [19], and the quasispecies model [20], which have been used to model social and multiagent network dynamics [21–23].

The emergence of cooperation within groups of selfish individuals, where cooperators compete with defectors, is an interesting research direction because it may seem to contradict natural selection. Studying the evolutionary stability in games involving cooperative and noncooperative behaviours such as the repeated prisoners' dilemma, a set of conditions were found which enable the emergence of cooperation and its evolutionary stability [24]. Recent results reveal that the evolution of strategies alone may be insufficient to fully exploit the benefits of cooperative behaviour and that coevolutionary rules can lead to a better understanding of the occurrence of cooperation [25].

Cyclic interactions, which occur in simple games such as rock-paper-scissors, emerge spontaneously in evolutionary games entailing volunteering, reward, and punishment and are common when the competing strategies are three or more regardless of the particularities of the game. Cyclic dominance is also a key to understanding predator-prey interactions or mating and growth strategies of biological organisms. A review of these issues [26] also presents an analysis of stability of spatial patterns that typically emerge in cyclic interactions.

Another study of the group interactions on structured populations including lattices, complex networks, and coevolutionary models highlights the synergic contribution of statistical physics, network science, and evolutionary game theory to the analysis of their dynamics [27]. In general, it is considered that group interactions cannot be reduced to the corresponding sum of pairwise interactions.

The evolution of public cooperation on complex networks is particularly important and has been studied, for example, in the context of public goods games [28] or the emergent behaviour of agent social networks [29].

In the context of diffusion, which allows players to move within the population, the analysis of the spatiotemporal patterns reveals the presence of chaos, which fits the complexity of solutions one is likely to encounter when studying group interactions on structured populations [30].

The applicability of the concept of evolutionary games can be found in social and natural sciences, with examples such as an RNA virus [31], ATP-producing pathways [32], and traffic congestion [33].

By using ideas from evolutionary computing, a multiagent system can be seen as a discrete Markov chain and its evolution as a Markov process, possibly with unknown transition probabilities [13]. In a model using this approach, in which the number of agents varies according to the fitness of the individuals, a definition for the degree of instability is proposed based on the entropy of the limit probabilities [34].

In the second category, some authors analysed the stability of small groups of fully interconnected particles [35–37]. A centralized algorithm for a system of particles that leads to irregular collapse and a distributed algorithm that leads to irregular fragmentation were proposed [38]. A universal definition of flocking for particle systems with similarities to Lyapunov stability was also suggested [39].

The stability of formation control was analysed in terms of the limitations of the number of communication channels, showing that stability is maintained by appropriately constructing the switching sequence of network structure [40]. Stable flocking motion can be obtained using a coordination scheme that generates smooth control laws for particles or agents, based on attractive and repulsive forces, and stability is investigated with double integrator dynamics [41]. The coordinated control of mobile robots where the agents are periodically interconnected leads to the formulation of a theoretical framework in which the stability of many distributed systems can be considered [42]. For the problem of formation control of a group of homogenous agents, necessary and sufficient conditions for stability in case of arbitrary time-invariant communication topologies were proposed, which reduced the analysis of multiagent systems with any number of agents to the analysis of a single agent [43].

Conditions for swarm stability of nonlinear high-order multiagent systems were also described based on the idea of space transformation, showing that swarm stability can be ensured by sufficient connectivity of graph topology and dissipative property regulated by relative Lyapunov function, with two independent variables, for time-varying or heterogeneous models [44].

Distributed control strategies for motion coordination were proposed and demonstrated to work for the rendezvous problem of planar roving agents using contraction theory. It was found that even if noise were present in the network, rendezvous would still be achieved with some final bounded error because of the contractivity of all the agents guaranteed by the protocol [45]. The rendezvous problem is related to the classical consensus problem in networks and therefore can be solved using a similar approach.

For systems that belong to the third category, stability was analysed for the problem of iterative contracting of tasks. The interactions were resource reallocations through a memoryless bidding protocol where every agent calculated its next bidding price using a discount drawn from a Gaussian distribution. The results showed that the network of agents

reached an equilibrium distribution, even in the presence of noise [46].

Another model of a network of agents interacting via time-dependent communication links was proposed which can be applied in domains such as synchronization, swarming, and distributed decision making. Each agent updates its current state based on the current information received from its neighbours. It was shown that, in case of bidirectional communication, convergence of the individual agents' states to a common value is guaranteed if, during each (possible infinite) time interval, each agent sends information to every other agent, either through direct communication or indirectly via intermediate agents. In case of unidirectional communication, convergence is proven if a uniform bound is imposed on the time it takes for the information to spread over the network [47].

In a scenario in which leaders are required to recruit teams of followers, satisfying some team size constraints, and where agents have only local information of the network topology, an algorithm was proposed and shown to converge to an approximate stable solution in polynomial time. For general graphs, it was found that there can be an exponential time gap between convergence to an approximate solution and convergence to a stable solution [48].

The problem of coordination in multiagent systems becomes more difficult when agents have asynchronous, uncertain clocks. For this situation, necessary and sufficient conditions for stability were suggested based on linear matrix inequalities. The analysis was applied to networked control with random sampling times, as well as an asynchronous consensus protocol. The stability of linear multiagent systems with noisy clocks was also studied [49].

In case of a supervisory control scheme that achieves either asymptotic stability or consensus for a group of homogenous agents described by a positive state-space model, necessary and sufficient conditions for the asymptotic stability, or the consensus of all agents, were derived under the positivity constraint [50].

Some approaches are hybrid, for example, trying to obtain stabilizing control laws for problems such as state agreement with quantized communication and distance-based formation control. Stabilizing control laws are provided when the communication graph is a tree [51].

For heterogeneous, interconnected multiagent systems where the interaction topology is an arbitrary directed graph, a general method was proposed to derive the transfer functions between any pair of agents with different dynamics. A class of multiagent systems is presented for which a separation principle is possible, in order to relate formation stability to interaction topology [52].

Another important issue when analysing the behaviour of a dynamical system is the presence of phase transitions. A phase transition in a system refers to the sudden change of a system property when some parameter of the system crosses a certain threshold value. This kind of changes has been observed in many fields, such as physics, for example, Ising magnetization model [53, 54], graph theory, for example, random graphs [55], cellular automata [56, 57], or biology, for example, the evolution of the numbers of two species [58]. Phase transitions have been observed in multiagent systems displaying simple communication behaviour, that is, when agents update their states based on the information about the states of their neighbours received under the presence of noise [59]. It was shown that, at a noise level higher than some threshold, the system generates symmetric behaviour or disagreement, whereas, at a noise level lower than the threshold, the system exhibits spontaneous symmetry breaking or consensus.

Our proposed multiagent system described in Section 3 also belongs to the third category but exhibits certain differences from other existing models.

3. Description of the Multiagent Interaction Protocol

The main goal in designing the structure and the interactions of the multiagent system was to find a simple setting that can generate complex behaviours. A delicate balance was needed in this respect. On the one hand, if the system is too simple, its behaviour will be completely deterministic. On the other hand, if the system is overly complex, it would be very difficult to assess the contribution of the individual internal components to its observed evolution. The multiagent system presented as follows is the result of many attempts of finding this balance. The wide range of observed behaviours from stable and periodic to chaotic and nondeterministically unstable was described in two previous papers [60, 61].

The proposed system is comprised of n agents; let A be the set of agents. Each agent has m needs and m resources, whose values lie in their predefined domains D_n, $D_r \subset \mathbb{R}^+$. This is a simplified conceptualization of any social or economic model, where the interactions of the individuals are based on some resource exchanges, of any nature, and where individuals have different valuations of the types of resources involved.

It is assumed that the needs of an agent are fixed (although it is possible to consider an adaptive mechanism [62, 63] as well), that its resources are variable, and that they change following the continuous interactions with other agents.

Also, the agents are situated in their execution environment: each agent a has a position π_a and can interact only with the other agents in its neighbourhood Λ_a. For simplicity, the environment is considered to be a bidimensional square lattice, but this imposes no limitation on the general interaction model; it can be applied without changes to any environment topology.

3.1. Social Model. Throughout the execution of the system, each agent, in turn, chooses another agent in its local neighbourhood to interact with. Each agent a stores the number of previous interactions with any other agent b, $i_a(b)$, and the cumulative outcome of these interactions, $o_a(b)$, which is based on the profits resulting from resource exchanges, as described in the following section.

When an agent a must choose another agent to interact with, it chooses the agent in its neighbourhood with the highest estimated outcome: $b^* = \arg\max_{b \in \Lambda_a} o_a(b)$.

The parallelism of agent execution is usually simulated by running the agents sequentially and in random order. Since one 3 of the goals of the system is to be deterministic, we define the execution order from the start. Thus, at any time, it can be known which agent will be active and with which other agent the active agent will choose to interact. In our case the random order is not necessary to generate complex behaviours. Even if the agents are always executed in lexicographic order (first A1, then A2, then A3, etc.), sudden changes in utilities still occur, although the overall aspect of the system evolution is much smoother.

3.2. Bilateral Interaction Protocol. In any interaction, each agent tries to satisfy the needs of the other agent as well as possible, that is, in decreasing order of its needs. The interaction actually represents the transfer of a resource quantum γ from an agent to the other. Ideally, each agent would satisfy the greatest need of the other.

For example, let us consider 3 needs (N) and 3 resources (R) for 2 agents a and b: $N_a = \{1, 2, 3\}$, $N_b = \{2, 3, 1\}$, $R_a = \{5, 7, 4\}$, $R_b = \{6, 6, 5\}$, and $\gamma = 1$. Since need 2 is the maximum of agent b, agent a will give b 1 unit of resource 2. Conversely, b will give a 1 unit of resource 3.

In order to add a layer of nonlinearity, we consider that an exchange is possible only if the amount of a resource exceeds a threshold level θ and if the giving agent a has a greater amount of the corresponding selected resource r_{sel} than the receiving agent b: $R_a(r_{\text{sel}}) > R_b(r_{\text{sel}})$ and $R_a(r_{\text{sel}}) > \theta$.

In the previous situation, if we impose a threshold level $\theta = 5$, agent a will still give b 1 unit of resource 2, but b will only satisfy need 1 for agent a.

Based on these exchanges, the resources are updated and the profit p_a is computed for an agent a as follows:

$$p_a = \gamma \cdot N_a\left(r_{\text{sel}}\right) \cdot \frac{R_b\left(r_{\text{sel}}\right)}{R_a\left(r_{\text{sel}}\right)}. \tag{1}$$

A bilateral interaction can bring an agent a profit greater than or equal to 0. However, its utility should be able to both increase and decrease. For this purpose, we compute a statistical average of the profit, p_{avg}, increase the utility of an agent if the actual profit is above p_{avg}, and decrease the utility if the profit is below p_{avg}.

Thus, the equation for updating the utility level of an agent a is

$$u_a \longleftarrow \frac{u_a \cdot i_a^{\text{adj}} + \eta \cdot \left(p_a - p_{\text{avg}}\right)}{i_a^{\text{adj}} + 1}, \tag{2}$$

where the adjusted number of interactions is $i_a^{\text{adj}} = \min(\sum_{b \in A} i_a(b), i_{\text{mem}})$, i_{mem} is the maximum number of overall interactions that the agent can "remember" (i.e., take into account), and η is the rate of utility change. At the beginning, the utility of the agent can fluctuate more, as the agent explores the interactions with its neighbours. Afterwards, the change in utility decreases but never becomes too small.

Similarly, the social outcome of an agent a concerning agent b is updated as follows:

$$o_a(b) \longleftarrow \frac{o_a(b) \cdot i_a(b) + \eta \cdot \left(p_a - p_{\text{avg}}\right)}{i_a(b) + 1}. \tag{3}$$

In this case, the social model concerns only 1 agent and thus the use of the actual number of interactions can help the convergence of the estimation an agent has about another.

Regarding the computation of the average profit, a global parameter of the system, we used a statistical approach where we took into account 1000 continuous interactions between two randomly initialized agents, which exchange resources for 100,000 time steps. The average profit depends on the number of resources, their domain, and the interaction threshold.

Since the social outcome depends on the average profit, the latter must be a fixed point of the corresponding system. If the initial value $p_{\text{avg}}^0 < p_{\text{avg}}$, the result of the simulation will provide a value $p_{\text{avg}}^{\text{sim}} > p_{\text{avg}}^0$ and vice versa. Therefore the correct value can be iteratively approximated by increasing or decreasing p_{avg}^0 until $p_{\text{avg}}^0 \approx p_{\text{avg}}^{\text{sim}} \approx p_{\text{avg}}$.

In the following sections, we will present some heuristic methods to stabilize the utility functions of the agents, such that they reach an equilibrium, in a minimally invasive way, that is, with the least amount of change to the system. We will explore the effect of small endogenous (internal) perturbations, in terms of alternative decisions made by the agents, to the stability of the multiagent system.

4. Interpretations of Stability

4.1. Statistical and Game Theoretical Interpretations. From a statistical point of view, the evolution of the multiagent system can be viewed as a Markov process with an initial distribution and transition matrix. The state of the system at time step n is represented by a random variable X_n, which is a vector that contains the probabilities of certain parameters that define the actual state of the system (e.g., agent locations, utilities, and characteristics). In this case, the system can be considered to be stable if the distribution of states converges to an equilibrium distribution; that is, $P(X_n = j) \to \pi_j$, when $n \to \infty$. In other words, the system is stable if the probability distribution of system states becomes independent of the time step n, for large values of n [13].

However, this interpretation cannot be easily applied for the proposed multiagent interaction protocol. First, the visible output of the system is continuous, but it could be discretized into a convenient number of states. Secondly, all the interactions in the system are deterministic; while it is possible to make a statistical interpretation of a larger number of behaviours, the probabilistic framework would generate only approximations of the real evolution of a particular configuration. Thirdly, the most important aspect is that the accumulation of the nonlinear effects makes it very difficult to estimate the transitions between different states, because these transitions do not depend on the decisions of one agent

at a time but on the aggregation of the decisions of all agents involved.

Stability is a relatively well-understood concept in physics, where it is regarded as a property of a stable equilibrium, that is, a state where the system returns on its own after a small perturbation. For multiagent systems, stability can also be defined with respect to perturbations of some sort, such as noise, variations in parameter values, or addition or removal of agents. If a small initial disturbance later becomes significant, the system is considered unstable. A multiagent system can be considered in equilibrium when the statistical properties of its performance indicators remain stationary when the external conditions that can impact the system vary [46].

In our case, it has been demonstrated that small perturbations can sometimes change the final state entirely, while in other cases the perturbations fade away [60, 61]. In the present paper, we use small perturbations to correct instability, such that the behaviour of the system should become easier to predict.

Multiagent systems can also be seen as a set of agents engaged in a multiplayer game. In game theory, stability is also a property of equilibrium, and the problem of finding an equilibrium is equivalent to the choice of an optimal strategy. Stability can then be used to describe the characteristics of the set of strategies in equilibrium, where the strategy of each player is the best response to the strategy of the others and where no player has any incentive to deviate, for example, Nash equilibrium [64].

Another related definition of stability considers the set of all actions performed by all agents in the system. An equilibrium point is one where no agent needs to further operate within the system and accepts the currently reached state. A set of actions, by the different agents, is stable if, assuming that an oracle could inform each agent about all the actions in the set performed by the other agents (and all the external events in the environment), each agent would do exactly what is in the set; that is, its observable behaviour would be exactly what the set predicts [65].

Considering again our multiagent system, the main difference in a game theoretic approach is that we take into account the observed behaviour and not the decisions that agents make. We need methods to find an equilibrium in an implicit not explicit way by considering the dynamic succession of interactions between agents.

Therefore, in the following sections, we try to identify some definitions of stability which are more appropriate for our multiagent system.

4.2. External Behaviour Interpretation. From the practical point of view, stabilization can be defined in a straightforward way as the lack of further change to the agent utilities, starting from a certain time step of the simulation (Figure 1).

As described in the previous works [60, 61], the multiagent system under analysis can exhibit many types of behaviours, such as being stable, periodical, chaotic, and nondeterministically unstable. However, in the rest of this paper, we will only distinguish between stable and unstable

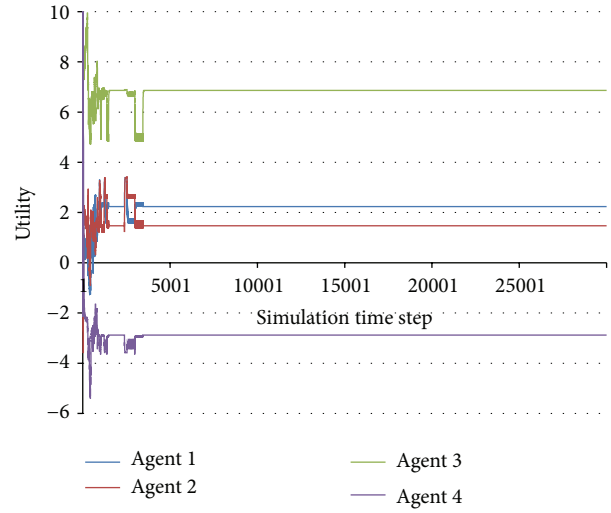

Figure 1: Stable behaviour.

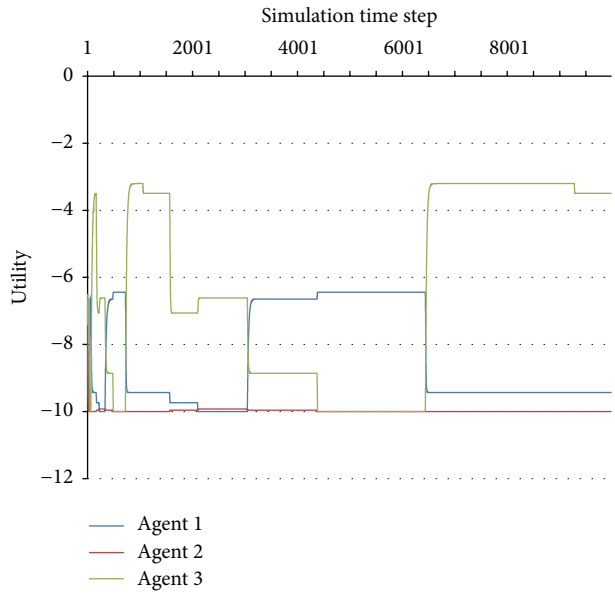

Figure 2: Unstable behaviour.

behaviours. Small periodic oscillations of the utility function will also be considered to represent stable behaviour.

When the system exhibits unstable behaviour (either deterministically chaotic or nondeterministically unstable), the utilities of the agents continue to exhibit changes throughout the simulation, with no clear horizon of becoming constant. An example of unstable behaviour is presented in Figure 2.

The stability of the system can be considered in a statistical way. A simple analysis has been made [61] showing that the probability of instability increases as the number of agents in the system increases. This is not surprising, because instability clearly depends on the complexity of agent interactions. However, when we consider the multiagent system at a lower level, we see that instability can occur even with as little as 3 agents; therefore, in the following, we aim at

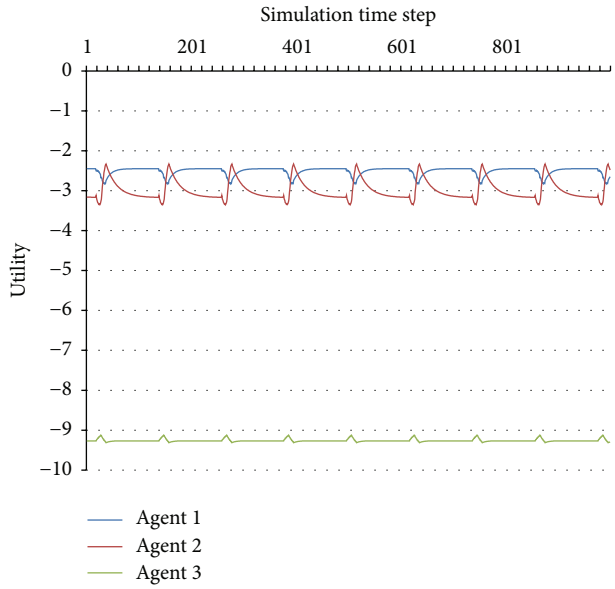

Figure 3: Unstable periodic behaviour.

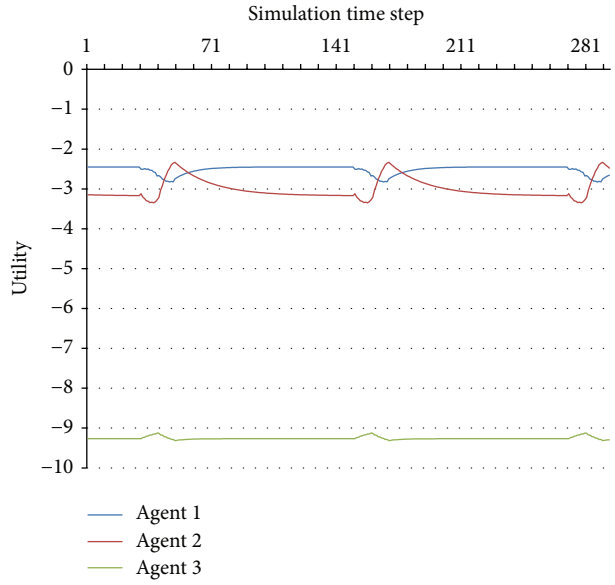

Figure 4: Close-up of the unstable periodic behaviour.

understanding the internal conditions that cause the overall instability.

From an external point of view, it is not difficult to distinguish between stability and instability. The procedure is the following. We first set an arbitrary number of simulation steps to analyse the behaviour. In a previous paper [61], we chose 10 million steps to observe whether an unstable behaviour eventually stabilizes, and it was found that it continues to exhibit sudden changes even after long periods of stability.

Then, since at the beginning of the simulation there is usually a transitory period with many changes, in considering the question of stability one should ignore this transitory unstable period.

In between these two time limits, one can observe the evolution of the utility functions. If the absolute difference between the minimum and maximum values of the utility function of the agents exceeds a certain threshold (e.g., 0.1 or 1), then the behaviour is considered to be unstable.

Another issue is the presence of periodic oscillations. Small oscillations are very often present in case of systems with a larger number of agents (e.g., above 6). In order to eliminate them and consider this type of behaviour stable, we can use a moving average window and smoothen the utility functions.

This method cannot be applied for complex periodic behaviours, such as the one presented in Figures 3 and 4. However, since this kind of behaviour is predictable, we will also consider it to be stable.

Thus, our focus for stabilization is on the aperiodic behaviours. Although with different probabilities, this type of unpredictable evolution can appear to any number of agents greater than 2. The reason why 2 agents cannot exhibit unstable behaviour is explained in the next section.

4.3. Internal Model Interpretation. While obviously useful, the external perspective of the system fails to address two fundamental questions of stability: why sudden changes can appear after long periods of stability and even if the system appears to be stable during the time period considered for analysis, what guarantees we have that it will not exhibit large changes again, sometime after the upper time limit of the simulation.

The answers to these questions can be found only by opening the "black box" of the internal social models of the agents.

Stability appears when the interactions of the agents follow a certain structure; for example, an agent a keeps on interacting with an agent b, because b brings a maximum profit. This situation is reflected in the social model of agent a, and thus a chooses b because b has the largest social outcome in agent a's internal model (according to (3)). If all the agents keep interacting with the same peers, the profits they receive (1) are the same in every time step and their utility functions quickly converge (2).

The utility function of an agent changes when the profit it receives in one step is different from the profit it has received in the previous step. This is caused by the nonlinearity in the way in which resources are given. A resource unit is given only when the amount that an agent has exceeds a certain threshold (e.g., 5). But when the agents keep interacting with the same peers in a stable way, the amount of resources an agent has at the beginning of a time step remains constant: a resource unit is passed between the agents in a neighbourhood until it gets back to the first agent.

However, the "opinion" an agent has about its peers is highly dynamic. The social outcome of the best agent to interact with can decrease. For a while that agent remains the best option. But at some point, the value of its social model decreases below the one of another agent. This causes a shift

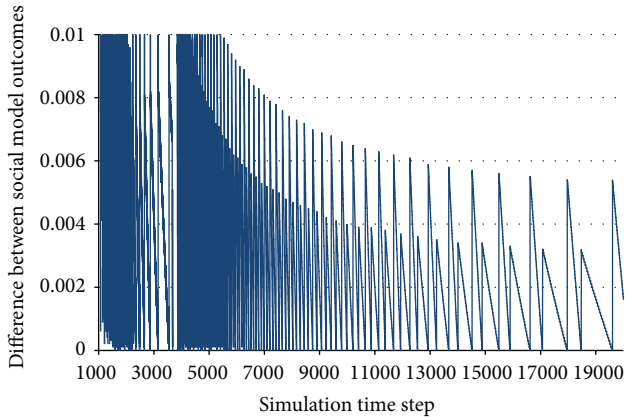

Figure 5: Internal changes in the social model leading to unstable behaviour.

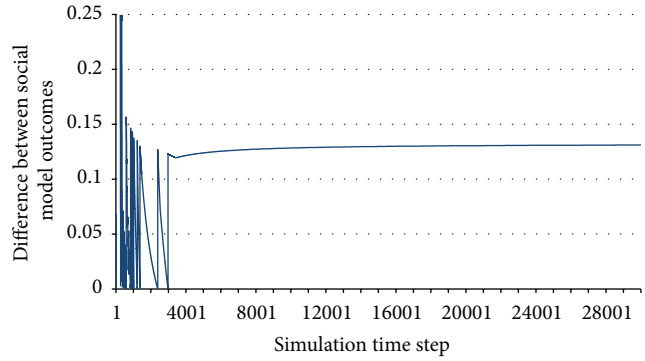

Figure 6: Internal changes in the social model leading to stable behaviour.

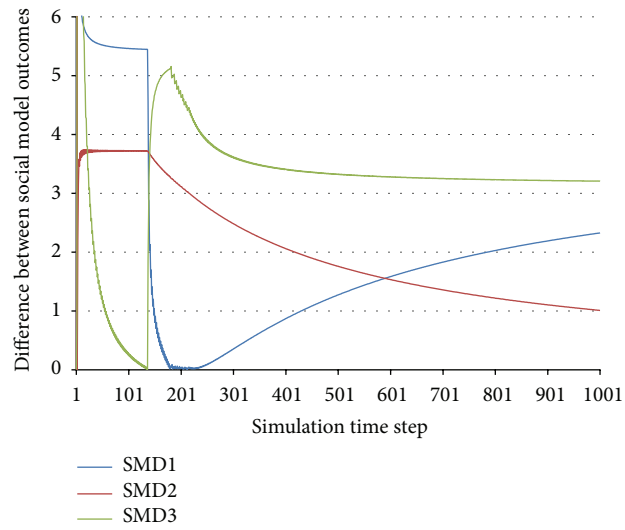

Figure 7: Interaction of changes in social model outcomes.

in the interaction pattern of an agent. The "second best" now becomes the "best." The interaction with this new agent can be better or worse than the interaction with the "old best." Some of its resources which the agent who initiates the interaction desires may be below the threshold and cannot be provided. This happens especially when the number of resources is small (e.g., below 3). Otherwise, agents can usually find a resource to give. But it can nevertheless bring about a lower profit to the receiver.

When the interaction is not profitable enough, the value of the social model of the "best option" again decreases, and soon another agent will be selected for interaction (either the previous best or a completely different one).

In this way, we can see that the changes in the social ranking of agents are the internal causes for instability.

In order to represent the internal changes more efficiently, especially when the number of agents is large, we can take into account the difference between the social outcome of the best peer and the social outcome of the second best. When the best becomes less attractive, the difference will decrease and a change will occur after it reaches 0. Let us consider the example with an unstable behaviour. We can see that all agents follow the same pattern of temporary stability and changes. Among them, Agent 4 has the lowest amplitude of these fluctuations. However, its internal model of the difference between its best and second best peer is displayed in Figure 5. We can see that the changes in all utility functions occur precisely when the difference in the social models of an agent reaches 0 and thus the agent's interaction patterns change. Although Agent 4 is not really affected by this phenomenon, the changes greatly impact the other agents. Basically, when an agent changes its preferred interaction partner, an external change occurs in its utility and/or the utility of other agents. If the ranks of the internal models keep changing, this will be reflected in the observed instability of the system.

On the other hand, let us now consider a situation with a stable behaviour. After some initial fluctuations in a transitory phase, an agent finds a peer with whom the interactions are rewarding. The difference in the social model outcomes between the best and the second best monotonously increases (although at some point it seems to converge), as displayed in Figure 6. The same thing happens to the other agents as well. In this case, we have a guarantee that the considered system is stable and will remain stable forever, since no changes can occur any longer.

As stated above, the swaps in any agent's social model can affect the system, and since they can simultaneously occur in more agents, especially in large systems, their effects can be combined in ways which are difficult to predict analytically. For example, let us consider a system with 3 agents, whose differences in the social model outcomes are displayed in Figure 7. As before, these values represent the difference between the social model outcome of the best peer, which seems to bring about the greatest profit to the considered agent, and the social model outcome of the second best peer. The 3 functions in the figure represent the way in which the difference varies for each of the 3 agents considered in this case.

One can notice two swaps that interact: the first in time step 136 for Agent 3 and the second, with certain oscillations,

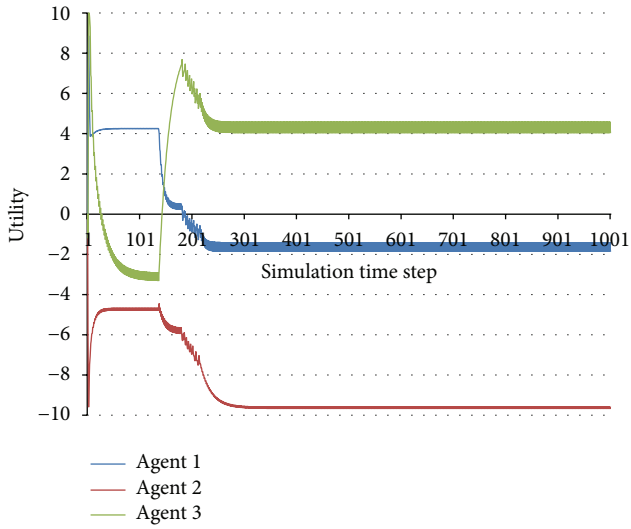

Figure 8: Observed behaviour when changes in the social model outcomes interact.

in time steps 183–229 for Agent 1. The consequence can be observed in Figure 8, especially in the utility functions of Agent 1 and Agent 2: their utility functions have a particular evolution until time step 136, a different one between 136 and 183, another oscillatory one between 183 and 229, and another shape after time step 229.

The internal perspective can provide satisfactory answers to the questions we introduced at the beginning of the section. Since the difference between the best and second best can decrease very slowly, we can have long periods of stability. However, once a swap occurs, the utility functions change and this can lead to highly unstable behaviour. Also, the swap can occur after the time limit of the simulation. The system will be stable for the considered period of time but could prove to be actually unstable if the analysed time period were extended. Of course, it could stabilize soon after, as well. Also, this is why the systems with only 2 agents are always stable: because each agent has only one social model and swaps cannot occur during execution.

The only guarantee identified so far is when the differences in the social models are nondecreasing. However, it was empirically observed that this rarely happens, especially in systems with a large number of agents. Therefore, we could consider that most systems are inherently unstable. From the practical point of view, the external stability for a (possibly extended) period of time remains a good way to assess stability. The main goal of stabilization would be to make the system stable in the "strong," internal sense; however since it may not be possible in general to do so, we can restrict the notion of stability to a "weaker" interpretation, dealing with the external observed behaviour.

4.4. Combining External and Internal Perspectives. In this section, we propose a method to check the stability of the multiagent system, taking into account both internal and external factors. The main idea is the following. We test whether the internal models of all the agents have a periodic evolution. If this is not the case, the system cannot be considered stable. Likewise, if the external evolution of the utility functions is not periodic, the system is not stable. If both the internal models of interactions and the external utility functions are periodic, we compute the period of the whole system as the least common multiple of all agents' periods. The overall period will be 1 if the system has converged into a stable state with constant utility functions. As stated before, by stability we also understand predictability, and thus a periodic system is considered under this assumption to be stable, although it is not constant. Finally, the differences in the social models outcomes are evaluated, and if the difference decreases for at least one agent, it is assumed that it will eventually reach 0 and cause a swap, and therefore the system is considered unstable. The method is presented in Pseudocode 1.

In Pseudocode 1, it is considered that an agent has a record of its recent history (e.g., the last 100 steps or more), in terms of the agents with which it has interacted, what resources have been exchanged, and the values of its utility function. PeriodicModel checks whether an agent has been involved in periodic interactions. PeriodicUtility checks whether the values of its utility function are periodic. In case of a periodic behaviour, modelPeriod and utilityPeriod contain the actual values. Since they can be different, the period of the overall system is assessed as the least common multiple of all periods of the individual agents, both internal and external. modelDifference represents the difference between the best social model outcome of an agent and the second best one. If this difference decreases, it will eventually become zero and will cause a swap and thus an unstable behaviour.

It is important to say that the test for stability should be applied only after some initial time steps, which constitute the transitory period. In the beginning, all systems show fluctuations and therefore this initial period is not relevant to assess stability. The length of the transitory period depends on the particular multiagent system but is usually longer when the system is comprised of more agents.

A less restrictive variant of the stability test is to consider only the external, observed behaviour. From the practical point of view, if the simulation has a finite number of steps, we could be interested in finding a way to predict the overall system performance only for the specified period of time. Thus, an external test for stability is presented in Pseudocode 2, which will be later used in two heuristic methods, when another criterion based on the hybrid test fails to be satisfied.

This procedure is actually a simplified version of Pseudocode 1. It no longer contains the internal perspective test which checks whether the internal models are periodic. Pseudocode 1 also takes into account the interaction history of the agents and if the model difference (between the best peer and the second best peer) decreases, the system is considered to be unstable. Pseudocode 2 only checks the external perspective, that is, whether the utilities of the agents are periodic. Constant utilities, the most commonly encountered case of stability, are also included in this category, with a period equal to 1.

```
procedure HybridStabilityTest
input: agents: the list of agents with a history of their internal models and utility function values; modelDifference: the difference
       between the best social model outcome and the second best social model outcome
output: stability (true or false) and the period of the overall behaviour if the system is stable
begin
    for each agent a in agents do
        if not PeriodicModel(a) then // internal perspective
            return (false, 0)
        end if
        if not PeriodicUtility(a) then // external perspective
            return (false, 0)
        end if
        for all time steps k in the interaction history of agent a do
            if modelDifference(a, k) − modelDifference(a, k − 1) < 0 then
                return (false, 0)
            end if
        end for
    end for
    periodList = empty list
    for each agent a in agents do
        add modelPeriod(a) to periodList
        add utilityPeriod(a) to periodList
    end for
    lcm = least common multiple of elements in periodList
    return (true, lcm)
end procedure
```

Pseudocode 1: Hybrid test for stability.

```
procedure ExternalStabilityTest
input: agents: the list of agents with a history of their utility function values
output: stability (true or false) and the period of the overall behaviour if the system is stable
begin
    for each agent a in agents do
        if not PeriodicUtility(a) then
            return (false, 0)
        end if
    end for
    periodList = empty list
    for each agent a in agents do
        add utilityPeriod(a) to periodList
    end for
    lcm = least common multiple of elements in periodList
    return (true, lcm)
end procedure
```

Pseudocode 2: External test for stability.

5. Difficulty of Predicting Stability

5.1. Classification Methods. In order to identify the nature of the multiagent system behaviours, we attempted to use classification methods to determine whether a system will be stable or unstable, based on its initial conditions. For this analysis, we used *Weka* [66], a popular collection of machine learning algorithms.

There are many classification algorithms available at the moment. For our analysis, we used four representative ones, which usually exhibit very good performance.

C4.5 [67] generates a decision tree by recursive partitioning of data. The algorithm considers all the possible attribute tests that can split the data within a node and chooses the test that gives the best information gain, that is, the partitioning that would result in the most homogeneous child nodes.

Table 1: Classification results for the stability of a multiagent system with 3 agents and 1 resource.

Algorithm	Training	Cross-validation
C4.5	63.2%	60.2%
Random Forest	98.8%	80.6%
*k*NN	100%	76.8%
SVM, RBF	95.6%	83%

Table 2: Classification results for the stability of a multiagent system with 10 agents and 10 resources.

Algorithm	Training	Cross-validation
C4.5	89.3%	64.6%
Random Forest	99.2%	68.4%
*k*NN	100%	74.2%
SVM, RBF	100%	74.2%
SVM, Poly2	100%	66.8%

It can handle both symbolic and numerical attributes. The algorithm supports tree pruning at the end of the training process, which cuts off some parts of the tree in order to avoid overfitting.

The random tree classifier builds a tree that considers k random features for each node and performs no pruning; therefore its error rate on the training set alone is rather small. A *random forest* [68] is composed of several random trees. To classify a new instance, the input vector is presented to each of the trees in the forest. Each tree provides a classification, and its decision is considered to be a "vote" for that class. The forest chooses the class with the most votes. The method tries to avoid overfitting by using bagging to build individual trees for slightly different data.

The *k-nearest neighbour* (*k*NN) algorithm reduces the learning effort by simply storing the instances and classifying new ones on the basis of their closeness to their "neighbours," that is, instances in the training set with similar attribute values. The nearest neighbour (NN) algorithm classifies a new instance in the same class as the closest stored instance in the attribute space. A straightforward extension is the *k*NN, where k neighbours, instead of 1, are taken into account. This approach is especially useful when the data are affected by noise and the nearest neighbour of an instance may indicate an erroneous class. Another optimization is to use distance weighting, that is, to give more importance to the neighbours which are closer to the instance to be classified.

Support vector machines [69] (SVM) are one of the best classification methods presently available. They consider the training instances as points in a multidimensional space, which can be transformed in order for the classes to become separable with a large margin. The transformation is performed by means of a function known as the kernel, for example, linear, polynomial, and radial basis functions. They also include a mechanism to reach a balance between good performance on the training set and the risk of overfitting, by allowing training instances to be misclassified by using a "soft margin," with the intention of providing a simpler model which could generalize better.

For the classification of stable or unstable systems, the simplest case is to consider 3 agents and only 1 resource. Consequently, each agent will have only 1 need. Using the test for stability introduced in Section 4.4, we generated a training dataset of 1000 instances corresponding to 560 stable and 440 unstable systems. The dataset was created by generating different systems with random initial conditions (i.e., the values for the resources and needs). In order to maximize the interactions, here and in all subsequent case studies, all the agents are considered to have the same position, although the general model allows them to be distributed in the environment and interact only with the peers in their neighbourhood.

The classification results, in terms of accuracy obtained for the training set and by using 10-fold cross-validation, are presented in Table 1.

We can notice that only the training set enjoys good performance. No algorithm proves good generalization capabilities. The decision tree cannot learn a good model even on the training set. Random Forest and *k*NN overfit the data. SVM has the best generalization performance, but it is still far from being satisfactory.

In another case study we considered 10 agents and 10 resources, and the training set contains 1000 instances with a certain degree of imbalance: 742 stable and 258 unstable systems. Table 2 shows the accuracy of the classification algorithms.

Except for C4.5, all the other techniques approximate the training set very well but perform poorly on cross-validation. Since the results of *k*NN and SVM with RBF kernel are identical, another type of kernel was also tried, the polynomial kernel of the second degree, which actually has lower generalization capability, although the error on the training set is 0. The reason for these results is that the data have a very high dimensionality (200 dimensions). Therefore, the two classes of training instances are very easily separable. However, this does not guarantee good generalization. It can be seen that *k*NN and SVM with RBF kernel have an error rate of 74.2%, which is exactly the percentage of stable instances. They actually fail to identify the characteristics of the unstable instances at all and classify them as being stable. The other algorithms perform even worse, with additional errors for the stable instances as well.

These results are not surprising, because the agents within the system are tightly coupled and the effect of their local interactions propagates throughout the simulation. Small changes accumulate and can affect the overall behaviour later on, in ways which are difficult or even impossible to predict beforehand. Moreover, since some behaviours of the system are chaotic [61], long-term prediction in such systems is impossible by definition. Therefore, we can conjecture that the stability (or instability) of the proposed interaction protocol is an emergent property and cannot be predicted only by analysing the initial static state of the agents.

We can see in Figure 9 the initial distribution of two resources for the case study with 3 agents. The resource of the third agent was omitted for simplicity. Each point in the figure

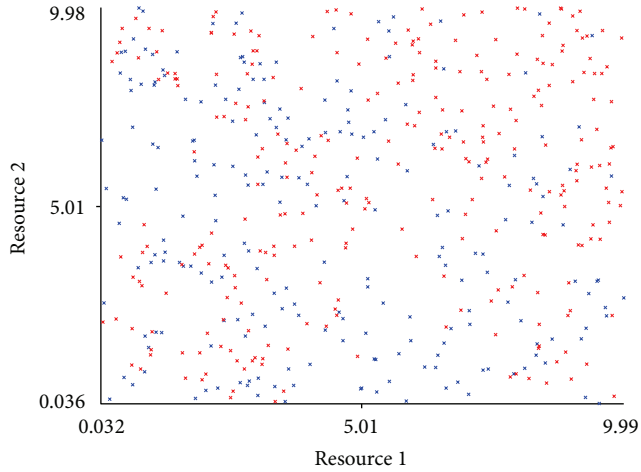

Figure 9: Initial distribution of two resources for stable and unstable instances.

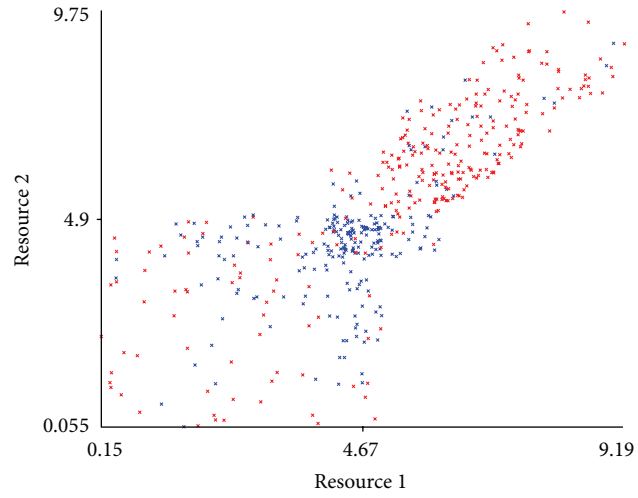

Figure 10: Final distribution of two resources for stable and unstable instances.

is an instance of the system where the coordinates represent the initial values of the agent resources. The red instances represent the systems that became stable during execution, and the blue ones represent unstable ones. The visual aspect of the stable and unstable instances is entirely random. This figure provides a helpful intuition that it may not be possible to predict the stability or instability of a system a priori, only by looking at its initial conditions.

It is also interesting to inspect the distribution of the two resources after the multiagent system has been executed for a number of time steps (e.g., 10000), as shown in Figure 10. One can see that the values of the resources tend to be balanced in cases where their value is higher than the exchange threshold, that is, 5. In this seemingly diagonal region, many systems are stable, because the agents have enough exchange choices in order to avoid the situation when an agent a receives a resource unit from agent b but is unable to give anything back to b. This would cause a drop in the profit of b and also in its social model of a, leading in time to a different choice of interaction and thus causing the instability of the system. Much more unstable instances are therefore placed very close to the exchange threshold. This could be a clear indicator of instability, but unfortunately this is a postfactum consequence of the simulation and cannot be predicted before its start.

5.2. Transition Points. Another line of investigation refers to the nature of phase transitions. Considering again a simple system with 3 agents and 1 resource, Figure 11 shows the stability (light grey) or instability (dark grey) of the system, based on the values of the resource. Unlike Figure 9, which only displayed sample points, Figure 11 displays a large number of initial values between 0 and 10, in order to identify regions of stability or instability. Since there are 3 values involved, we chose to represent 3 separate graphs, in which the abscissa corresponds to the resource of Agent 1 and the ordinate to the resource of Agent 2, and each graph corresponds to one of the following values of the resource of Agent 3: 0, 5, and 10, respectively. The graphs are not entirely symmetric because the needs of the agents are different.

Then, we tried to zoom in as closely as possible to the border between a stable region and an unstable region. Using the dichotomy technique, we could identify the values of a single resource for which the system would be stable (v_s) or unstable (v_u), with $|v_s - v_u| < 10^{-6}$. It was observed that the multiagent system does not exhibit gradual phase transitions. The behaviour of the system is quite similar when the resource value is either v_s or a stable value farther away from the border, compared to a resource value of either v_u or an unstable value farther away from it. Only the time steps when the swaps occur in case of instability can be somewhat different.

We can observe this change in the situation with 3 agents and 1 resource. We can use the sum of utilities as a reference, to have a simpler image of the system behaviour. Figure 12 shows the difference between two cases given by a change of only 10^{-6} in the resource of one agent, lying on both sides of the stability border and very close to it. One can notice that on the initial part of the simulation the difference is 0, but at some point the difference in resources causes an agent to make an alternative decision, and from then on the overall behaviour of the two systems becomes very different.

Let us consider now the same system configuration and see what happens when we make small changes in the resource of one agent, so as to cause the system to traverse a succession of stable–unstable–stable behaviours. In Figure 13, the ordinate represents the maximum difference between the sums of utility functions in the initial stable behaviour and in the situation where a resource is changed by the amount represented on the abscissa. More clearly, by changing the initial resource amount of one agent, we can cause the system to be stable or unstable. This is similar to traversing one of the subfigures in Figure 11 horizontally. At first, the system proves to be stable. In a subsequent run, when the resource amount is increased, it may enter an unstable area. By further increasing the resource, the system instance moves farther

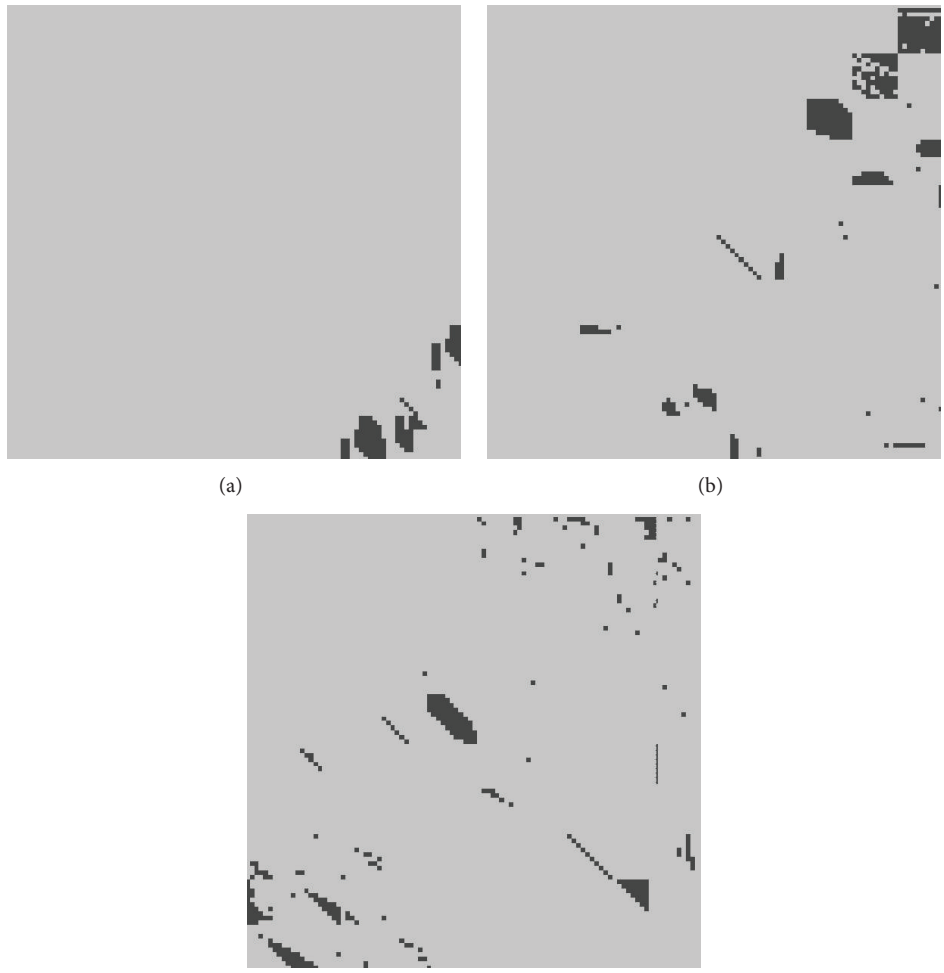

Figure 11: Continuous representation of stable and unstable systems in terms of resources R_1 and R_2, where (a) $R_3 = 0$; (b) $R_3 = 5$; and (c) $R_3 = 10$.

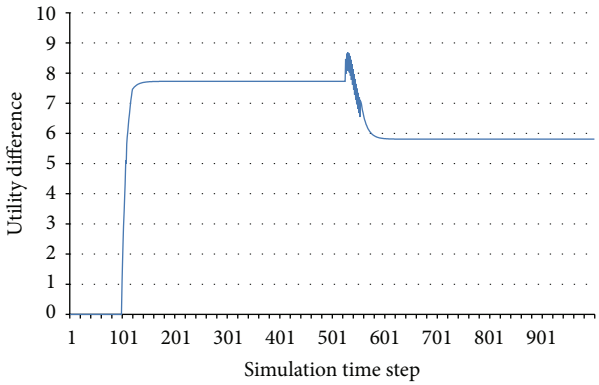

Figure 12: Difference between the sums of utilities with 10^{-6} difference in the resource of one agent.

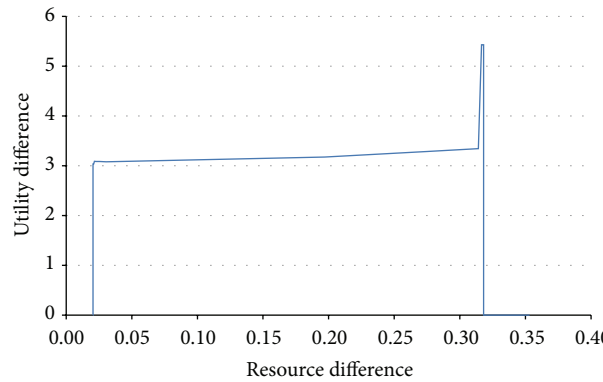

Figure 13: Maximum difference in a stable behaviour while traversing stable and unstable regions.

to the right and may enter a stable area again. We know how the sum of utilities varies in the initial, unchanged configuration. For each run, we also know the sum of utility functions. For each time step of the simulation, we can assess the difference between the sum of utilities in the unchanged system and the sum of utilities in the changed system. Figure 13 represents the maximum of those values, considering the entire simulation length. We can notice

the sudden changes that appear when crossing the stability border, that is, when the difference in resource is around 0.02 and around 0.32.

By studying many similar situations, it was observed that the phase transitions of the multiagent system are abrupt, and changes between stable and unstable behaviours can be caused by minute changes in the internal states of the agents.

6. Stabilization Methods

Since our empirical analysis shows that it is impossible to find out whether a system with a given configuration will be stable or not, so as to be able to select only the stable ones, we propose three heuristic methods for the stabilization of a multiagent system during its execution.

6.1. Hybrid Model Stabilization. The first stabilization method, presented in Pseudocode 3, takes into account the "strong" criterion for stability, based on both internal models and external behaviour. By using the HybridStabilityTest procedure, it is assumed that we have identified an unstable behaviour and we apply the method only in this case. It is applied during the simulation after the transitory period and consists in small changes being made to the internal models of the agents.

The basic idea of the method is that if the difference of the social model outcomes decreases, it will eventually become 0 and will cause a swap. Therefore we can anticipate it and change the internal social model of the agent, in order to make it choose its second best peer sb for interaction. This alternative decision can be the right one and in this case the social model of sb will start to increase and the chance of stabilization will increase as well. Or it can be a poor decision and sb will provide less profit for the agent. In this case, given the small resulting difference between the social model outcomes of the two peers, eps (which can be, e.g., 10^{-6}), if the second best is not good enough, the mistaken alternative decision will be quickly corrected.

Here and also in the next method, minDifference is one of the parameters which controls the moment when the actual perturbation of alternative decision making takes place. A value of 0 leads to very rare changes. The greater the minDifference, the higher the frequency of perturbations. However, a balance should be reached between the probability of stabilization and the simultaneous goal of achieving it in a minimally invasive way. It was observed in general that a higher value leads to a slight increase in the stabilization probability. For the following case studies, a value of 10^{-3} was used. Another parameter employed to control the number of changes is minStepsToChange. We used a value of 100 in the subsequent case studies, meaning that an agent model is swapped only if there have not been any other swaps affecting that agent in the previous 100 time steps.

6.2. Preemptive External Stabilization. The second stabilization method relies on the observation that many multiagent systems following the proposed protocol do not reach an internal stabilization but can nevertheless behave in a stable way for the duration of the simulation (and, in any case, for very long periods of time, if the difference between the social models outcomes decreases very slowly). The method is presented in Pseudocode 4.

The method implements an answer to the classical question "what if." Since our multiagent systems are completely deterministic, we can simulate their evolution into the future, starting from any current state of the agents. Its basic idea is to verify that the system remains externally stable for some desired period of time into the future. Unlike the internal models, whose difference could provide some indications about possible swaps, the external behaviour must be observed for a prolonged period of time, to see if the system remains stable, because, as already stated, sudden changes could always appear after long periods of seeming stability. Observing the past behaviour is not enough to assess the future behaviour.

The method, which is more computationally intensive than the previous one, first makes a copy of the current state of the environment (including all the agents) and continues the simulation in this virtual environment for a number of desired time steps (e.g., 1000 or 10000). The endogenous perturbation mechanism is the same as in the first method. In every step of the simulation, or every 10 steps, to make it faster, the effects of the perturbation are verified. If the utility functions of the agents do not become constant or periodic in the virtual simulation, another change is made and so on.

The main *for* block in Pseudocode 4 is the same as in Pseudocode 3. The difference is that the test for stability is performed in the virtual environment, and the changes are applied only when instability is detected there.

6.3. Random External Stabilization. If the methods presented above fail, we can apply a more invasive way of stabilization. This method, presented in Pseudocode 5, is similar in a way to the mutation in evolutionary algorithms, which can get a solution out of a local optimum. Its basic idea is to reset the social models of the agents with a small probability in order to allow the creation of a new structure of interactions.

The same technique of checking forward the stability of the solution is applied. However, the perturbation mechanism is much more simple, because it does not take into account the social models of the agents. The social model outcomes of the agents are randomly reinitialized, and this gives them a chance to interact with different peers which may in turn lead to stabilization.

The test for external stability is the same as in Pseudocode 4; that is, it is performed in a virtual environment. However, the perturbation procedure is completely different, because it does not rely on the internal models, but it is randomly applied with a small probability.

6.4. Results and Discussion. In this section we assess the performance of the three proposed methods. Given their different nature, they are applied sequentially, trying the next only when the one before it, implementing a stronger stability criterion, fails. First, we attempt to stabilize the system using

```
procedure HybridModelStabilization
input: step: current simulation step, agents: the list of agents; maxDifference: a threshold for social model outcome difference;
       minStepsToChange: the minimum number of simulation steps between changes; eps: a very small number;
       lastChange: a list with the last simulation step when a change was made to the internal model of an agent;
       noChanges: the number of changes made to the system so far
output: updated noChanges
begin
    for each agent a in agents do
        difference = modelDifference(a, step) − modelDifference(a, step − 1)
        if difference < maxDifference and step − lastChange(a) > minStepsToChange then
            a.SocialModels(a.FirstChoice).Outcome = a.SocialModels(a.SecondChoice).Outcome − eps
            lastChange(a) = step
            noChanges = noChanges + 1
        end if
    end for
    return noChanges
end procedure
```

Pseudocode 3: Hybrid model stabilization.

```
procedure PreemptiveExternalStabilization
input: step: current simulation step, agents: the list of agents; maxDifference: a threshold for social model outcome difference;
       minStepsToChange: the minimum number of simulation steps between changes; lastChange: a list with the last
       simulation step when a change was made to the internal model of an agent; environment: the environment of the agents;
       virtSimSteps: number of steps to run the virtual simulation; eps: a very small number; noChanges: the number of changes
       made to the system so far
output: updated noChanges
begin
  virtualEnvironment = Copy(environment)
  RunSimulation(virtualEnvironment, virtSimSteps)
  (stable, lcm) = ExternalStabilityTest(virtualEnvironment)
  if not stable then
    for each agent a in agents do
        difference = modelDifference(a, step) − modelDifference(a, step − 1)
        if difference < maxDifference and step − lastChange(a) > minStepsToChange then
            a.SocialModels(a.FirstChoice).Outcome = a.SocialModels(a.SecondChoice).Outcome − eps
            lastChange(a) = step
            noChanges = noChanges + 1
        end if
    end for
  end if
  return noChanges
end procedure
```

Pseudocode 4: Preemptive external stabilization.

the hybrid criterion, which guarantees absolute stability. If it is not possible, we try a principled way to stabilize the external behaviour. If this also fails, we apply random perturbations in order to cause a different set of interactions which may bring about stability.

For the results presented below, 100 unstable multiagent systems were considered for each scenario, with different numbers of agents (3, 6, and 10) and resources (1, 2, 5, and 10). The success probability of a method was recorded. It must also be said that a small percentage of systems (less than 1%) could not be stabilized by running the three methods sequentially. However, they could be stabilized quite quickly by a second run of the third method.

In each scenario, the average number of steps after which the stabilization occurs (out of 10000 total time steps) and the average number of changes (as indicated by the noChanges variable in the pseudocode of the methods) were computed. The number of changes indicates the number of consecutive interventions during a single run before the system stabilizes. In most of the considered cases, the stabilized systems had constant values for the utility functions (period 1), and few exhibited a period of 2.

```
procedure RandomExternalStabilization
input: agents: the list of agents; environment: the environment of the agents; virtSimSteps: number of steps to run the virtual
       simulation; noChanges: the number of changes made to the system so far
output: updated noChanges
begin
  virtualEnvironment = Copy(environment)
  RunSimulation(virtualEnvironment, virtSimSteps)
  (stable, lcm) = ExternalStabilityTest(virtualEnvironment)
  if not stable then
    for each agent a in agents do
      if Random(0, 1) < 0.1 then // with probability 0.1
        for each agent b in agents such as b ≠ a do
          a.SocialModels(b).Outcome = Random(0, 1)
          a.SocialModels(b).NoInteractions = 1
          noChanges = noChanges + 1
        end for
      end if
    end for
  end if
  return noChanges
end procedure
```

Pseudocode 5: Random external stabilization.

Table 3: Performance of the hybrid model stabilization method.

Agents	Resources	Success probability	Avg. steps	Avg. changes
3	1	82	248.95	4.42
3	2	30	522.98	8.29
3	5	14	754.93	12.11
3	10	9	677.56	6.72
6	1	27	1623.74	56.27
6	2	37	2168.37	68.65
6	5	41	1775.72	57.27
6	10	8	1242.56	36.64
10	1	14	2279.35	193.60
10	2	13	2977.03	174.57
10	5	11	4765.13	353.33
10	10	11	1700.17	265.45

Table 4: Performance of the preemptive external stabilization method.

Agents	Resources	Success probability	Avg. steps	Avg. changes
3	1	13	385.37	2.83
3	2	38	927.39	11.92
3	5	36	1167.44	13.94
3	10	12	1164.23	21.42
6	1	68	44.14	11.78
6	2	61	311.97	19.55
6	5	58	1026.18	32.88
6	10	75	1610.95	51.36
10	1	85	261.04	15.55
10	2	86	676.85	40.97
10	5	88	1444.33	65.12
10	10	88	1792.74	95.90

Tables 3, 4, and 5 display these results for each of the three methods.

If we consider the first scenario, with 3 agents and 1 resource, we can see that the first method is very effective. In fact, it is the only situation in which most systems can be internally stabilized. The probability of success of the first method decreases as the number of resources decreases. It can be noticed that the systems with 3 agents prove rather difficult to stabilize, once they settled into a structure of interactions. Actually, the agents have a very limited number of choices, since they can only decide between 2 peers. Consequently, only method 3 succeeds in stabilizing them for a larger number of resources. As the complexity of the systems increases, the second method becomes increasingly successful. These systems are difficult to stabilize internally, but they display more flexibility in terms of the number of choices of interaction an agent has. Most systems with 10 agents could be stabilized using the preemptive external method, while the random method was virtually no longer necessary.

One can also see that method 3 is less efficient in terms of stabilization speed. The average number of steps after which the systems become stable is the greatest out of the three methods, and also the number of changes to the system is the greatest. Since it relies on random perturbations rather than a definite principle of stabilization, different changes are tried until the system eventually stabilizes. This is why it is used as

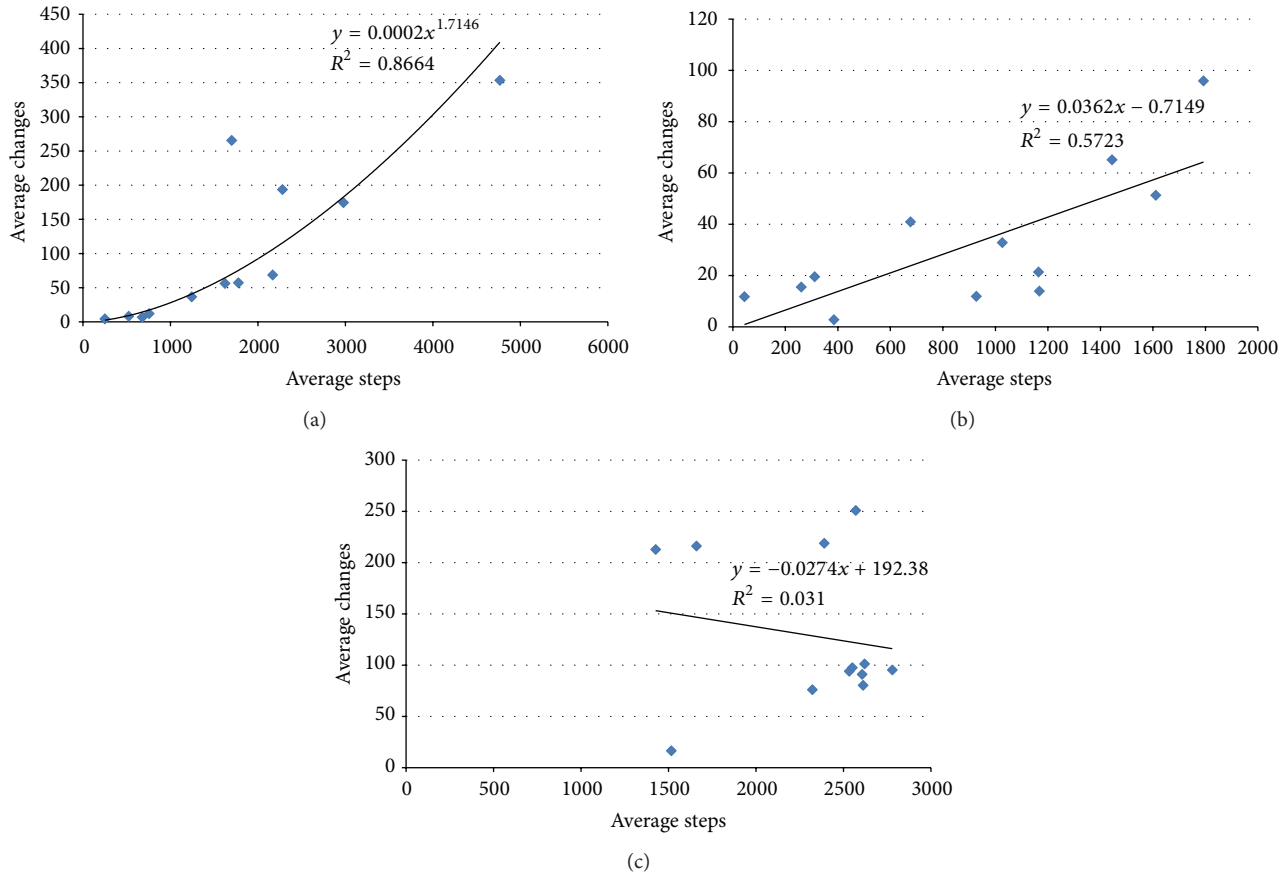

Figure 14: The variation of the number of changes with the number of steps until stabilization: (a) hybrid model; (b) preemptive external; and (c) random external.

Table 5: Performance of the random external stabilization method.

Agents	Resources	Success probability	Avg. steps	Avg. changes
3	1	5	1515.72	16.58
3	2	32	2322.16	76.12
3	5	50	2611.80	80.47
3	10	79	2606.61	91.09
6	1	5	2532.47	94.14
6	2	2	2779.04	95.38
6	5	1	2550.06	97.64
6	10	17	2619.98	101.32
10	1	1	1426.30	212.93
10	2	1	1659.65	216.18
10	5	1	2389.85	218.87
10	10	1	2569.67	250.91

the last resort for stabilization, when the other two methods fail.

Comparing the first two methods, we can see that they are similar in terms of average steps needed for stabilization. The second method seems more efficient though in terms of the number of changes to the system. It is also interesting to view the average number of changes as a function of the average number of steps needed for stabilization (Figure 14). For methods 1 and 2, there seems to be a relation between the two output measures; however, a relation does not seem to exist for the third, random method. The best fit corresponds to the first method, but the number of changes does not scale well when the number of steps needed for stabilization increases. The second method seems to be more efficient from this point of view, as well, even if there are quite large differences between the curve fitting line and the actual data.

7. Conclusions

The main contributions of this paper are the identification of the internal cause of instability in the multiagent systems following the proposed interaction protocol and the design of three methods that can be used to stabilize unstable systems. However, they are heuristic, and they can be applied successively, trying to satisfy stronger criteria first and ending with random perturbations, which are less efficient and more invasive but useful in case of less flexible configurations. Future work should investigate whether an algorithmic solution exists for this problem.

Another important issue is to identify an analytic model of phase transitions. Phase transitions are often studied in physical systems [70]; however, they have been demonstrated to occur in other agent-based systems, as well [71]. So far, there is only empirical evidence about the abrupt nature of phase transition in the multiagent systems considered in our paper. A formal proof would be needed as the next step. Also, it would be interesting to study whether one can identify a complexity index such as the rate of population growth in the logistic map and to see, based on this index, if patterns emerge when complexity increases, such as Feigenbaum constant which appears in the logistic map and some fractals, or other complexity patterns in cellular automata. In our case, such an index may likely depend on the number of agents, which adds an additional layer of difficulty to the problem.

Our research can be applied in agent-oriented computing and simulation. In a multiagent memetic system, similar to that presented in [72], which involves the exchange of resources among agents, the amount of resources gathered by the agents can become a basis for the evaluation of the agents. This can lead to the decision of reproduction or termination, allowing the inheritance of the genotype of agents in the system. Another application can be the modelling or enhancing of agent-based simulation of cooperation in a population of agents interacting in a spatial domain, which exchange resources and compete among themselves, for example, by playing the iterated prisoners' dilemma game [73].

Other applications could be found in social or economic simulations or for the modelling of biological processes. The concepts of stability and equilibrium are crucial aspects in all these domains, and therefore the idea of controlling such systems with small perturbations can have a great potential. The heuristics presented in this work are designed for deterministic complex systems; otherwise the stability test in a virtual environment which simulates future behaviour would be inapplicable. Another interesting investigation would be to study whether they can still be used for a more general class of multiagent systems, where determinism is not guaranteed, but shortterm predictions can still be made in an approximate form.

Conflict of Interests

The author declares there is no conflict of interests regarding the publication of this paper.

References

[1] M. Wooldridge, *An Introduction to MultiAgent Systems*, John Wiley & Sons, 2002.

[2] W. Balzer, "SMASS: a sequential multi-agent system for social simulation," in *Tools and Techniques for Social Science Simulation*, R. Suleiman, K. G. Troitzsch, and N. Gilbert, Eds., pp. 65–82, Physica, Heidelberg, Germany, 2000.

[3] A. U. Frank, S. Bittner, and M. Raubal, "Spatial and cognitive simulation with multi-agent systems," in *Spatial Information Theory*, D. R. Montello, Ed., vol. 2205 of *Lecture Notes in Computer Science*, pp. 124–139, Springer, Berlin, Germany, 2001.

[4] H. Hexmoor, S. G. Venkata, and D. Hayes, "Modelling social norms in multiagent systems," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 18, no. 1, pp. 49–71, 2006.

[5] R. Sun, "Cognitive architectures and multi-agent social simulation," in *Multi-Agent Systems for Society*, D. Lukose and Z. Shi, Eds., vol. 4078 of *Lecture Notes in Computer Science*, pp. 7–21, Springer, Berlin, Germany, 2009.

[6] J. E. Almeida, R. Rosseti, and A. L. Coelho, "Crowd simulation modeling applied to emergency and evacuation simulations using multi-agent systems," in *Proceedings of the 6th Doctoral Symposium on Informatics Engineering (DSIE '11)*, pp. 93–104, Porto, Portugal, 2011.

[7] L. An, "Modeling human decisions in coupled human and natural systems: review of agent-based models," *Ecological Modelling*, vol. 229, pp. 25–36, 2012.

[8] M. D. A. Rounsevell, D. T. Robinson, and D. Murray-Rust, "From actors to agents in socio-ecological systems models," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 367, no. 1586, pp. 259–269, 2012.

[9] M. Birkin and B. Wu, "A review of microsimulation and hybrid agent-based approaches," in *Agent-Based Models of Geographical Systems*, A. J. Heppenstall, A. T. Crooks, T. Andrew, L. M. See, and M. Batty, Eds., pp. 51–68, Springer, Dordrecht, The Netherlands, 2012.

[10] A. Morris, W. Ross, H. Hosseini, and M. Ulieru, "Modelling culture with complex, multi-dimensional, multi-agent systems," in *Perspectives on Culture and Agent-Based Simulations, Integrating Cultures*, V. Dignum and F. Dignum, Eds., vol. 3 of *Studies in the Philosophy of Sociality*, pp. 13–30, Springer, 2014.

[11] C. Castelfranchi, "Making visible 'the invisible hand': the mission of social simulation," in *Interdisciplinary Applications of Agent-Based Social Simulation and Modeling*, D. F. Adamatti, G. P. Dimuro, and H. Coelho, Eds., pp. 1–19, IGI Global, 2014.

[12] A. L. C. Bazzan and F. Klügl, "A review on agent-based technology for traffic and transportation," *The Knowledge Engineering Review*, vol. 29, no. 3, pp. 375–403, 2014.

[13] M. Chli, P. de Wilde, J. Goossenaerts et al., "Stability of multi-agent systems," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, E. Santos Jr. and P. Willett, Eds., vol. 1, pp. 551–556, IEEE, Piscataway, NJ, USA, October 2003.

[14] J. M. Smith, *Evolution and the Theory of Games*, Cambridge University Press, Cambridge, UK, 1982.

[15] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, Princeton, NJ, USA, 1947.

[16] C. Castellano, S. Fortunato, and V. Loreto, "Statistical physics of social dynamics," *Reviews of Modern Physics*, vol. 81, no. 2, pp. 591–646, 2009.

[17] J. Hofbauer and K. Sigmund, *The Theory of Evolution and Dynamical Systems*, Cambridge University Press, Cambridge, UK, 1988.

[18] P. D. Taylor and L. B. Jonker, "Evolutionarily stable strategies and game dynamics," *Mathematical Biosciences*, vol. 40, no. 1-2, pp. 145–156, 1978.

[19] K. M. Page and M. A. Nowak, "Unifying evolutionary dynamics," *Journal of Theoretical Biology*, vol. 219, no. 1, pp. 93–98, 2002.

[20] M. Eigen, "Selforganization of matter and the evolution of biological macromolecules," *Die Naturwissenschaften*, vol. 58, no. 10, pp. 465–523, 1971.

[21] S. A. di Collobiano, K. Christensen, and H. J. Jensen, "The tangled nature model as an evolving quasi-species model," *Journal of Physics A: Mathematical and General*, vol. 36, no. 4, pp. 883–891, 2003.

[22] R. Olfati-Saber, "Evolutionary dynamics of behavior in social networks," in *Proceedings of the 46th IEEE Conference on Decision and Control (CDC '07)*, pp. 4051–4056, New Orleans, Lo, USA, December 2007.

[23] C. Hauert, "Replicator dynamics of reward & reputation in public goods games," *Journal of Theoretical Biology*, vol. 267, no. 1, pp. 22–28, 2010.

[24] M. A. Nowak, A. Sasaki, C. Taylor, and D. Fudenherg, "Emergence of cooperation and evolutionary stability in finite populations," *Nature*, vol. 428, no. 6983, pp. 646–650, 2004.

[25] M. Perc and A. Szolnoki, "Coevolutionary games—a mini review," *BioSystems*, vol. 99, no. 2, pp. 109–125, 2010.

[26] A. Szolnoki, M. Mobilia, L. Jiang, B. Szczesny, A. M. Rucklidge, and M. Perc, "Cyclic dominance in evolutionary games: a review," *Journal of the Royal Society Interface*, vol. 11, no. 100, Article ID 20140735, 2014.

[27] M. Perc, J. Gómez-Gardeñes, A. Szolnoki, L. M. Floría, and Y. Moreno, "Evolutionary dynamics of group interactions on structured populations: a review," *Journal of the Royal Society Interface*, vol. 10, no. 80, 2013.

[28] F. C. Santos, M. D. Santos, and J. M. Pacheco, "Social diversity promotes the emergence of cooperation in public goods games," *Nature*, vol. 454, no. 7201, pp. 213–216, 2008.

[29] F. Leon, "Emergent behaviors in social networks of adaptive agents," *Mathematical Problems in Engineering*, vol. 2012, Article ID 857512, 19 pages, 2012.

[30] J. Y. Wakano and C. Hauert, "Pattern formation and chaos in spatial ecological public goods games," *Journal of Theoretical Biology*, vol. 268, pp. 30–38, 2011.

[31] P. E. Turner and L. Chao, "Prisoner's dilemma in an RNA virus," *Nature*, vol. 398, no. 6726, pp. 441–443, 1999.

[32] T. Pfeiffer, S. Schuster, and S. Bonhoeffer, "Cooperation and competition in the evolution of ATP-producing pathways," *Science*, vol. 292, no. 5516, pp. 504–507, 2001.

[33] M. Perc, "Premature seizure of traffic flow due to the introduction of evolutionary games," *New Journal of Physics*, vol. 9, article 1, 2007.

[34] P. de Wilde and G. Briscoe, "Stability of evolving multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 41, no. 4, pp. 1149–1157, 2011.

[35] Y. Liu, K. M. Passino, and M. M. Polycarpou, "Stability analysis of M-dimensional asynchronous swarms with a fixed communication topology," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 76–95, 2003.

[36] E. W. Justh and P. S. Krishnaprasad, "A simple control law for UAV formation flying," Tech. Rep. 2002-38, Institute for Systems Research, University of Maryland, College Park, Md, USA, 2002, http://www.isr.umd.edu/~krishna/images/formation_flying_TR_2002-38.pdf.

[37] V. Gazi and K. M. Passino, "Stability analysis of swarms," *IEEE Transactions on Automatic Control*, vol. 48, no. 4, pp. 692–697, 2003.

[38] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, "Stable flocking of mobile agents. Part II. Dynamic topology," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 2, pp. 2016–2021, Maui, Hawaii, USA, December 2003.

[39] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401–420, 2006.

[40] G. Mohanarajah and T. Hayakawa, "Formation stability of multi-agent systems with limited information," in *Proceedings of the American Control Conference (ACC '08)*, pp. 704–709, Seattle, Wash, USA, June 2008.

[41] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, "Stable flocking of mobile agents, Part I: fixed topology," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, vol. 2, pp. 2010–2015, Maui, Hawaii, USA, December 2003.

[42] B. Goodwine and P. Antsaklis, "Multiagent compositional stability exploiting system symmetries," Tech. Rep. ISIS-2012-004, ISIS Group, University of Notre Dame, Notre Dame, Ind, USA, 2012, http://www3.nd.edu/~isis/techreports/isis-2012-004.pdf.

[43] A. Popov and H. Werner, "Robust stability of a multi-agent system under arbitrary and time-varying communication topologies and communication delays," *IEEE Transactions on Automatic Control*, vol. 57, no. 9, pp. 2343–2347, 2012.

[44] N. Cai, J.-W. Cao, H.-Y. Ma, and C.-X. Wang, "Swarm stability analysis of nonlinear dynamical multi-agent systems via relative Lyapunov function," *Arabian Journal for Science and Engineering*, vol. 39, no. 3, pp. 2427–2434, 2014.

[45] G. Russo and M. di Bernardo, "Solving the rendezvous problem for multi-agent systems using contraction theory," in *Proceedings of the 48th IEEE Conference on Decision and Control Held Jointly with the 28th Chinese Control Conference*, pp. 5821–5826, Shanghai, China, December 2009.

[46] L. C. Lee, H. S. Nwana, D. T. Ndumu, and P. De Wilde, "The stability, scalability and performance of multi-agent systems," *BT Technology Journal*, vol. 16, no. 3, pp. 94–102, 1998.

[47] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.

[48] L. Coviello and M. Franceschetti, "Distributed team formation in multi-agent systems: stability and approximation," in *Proceedings of the 51st IEEE Conference on Decision and Control (CDC '12)*, pp. 2755–2760, Maui, Hawaii, USA, December 2012.

[49] A. Lamperski and A. Papachristodoulou, "Stability and consensus for multi-agent systems with Poisson clock noise," in *Proceedings of the IEEE 53rd Annual Conference on Decision and Control*, pp. 3023–3028, IEEE, Los Angeles, Calif, USA, December 2014.

[50] M. E. Valcher and P. Misra, "On the stabilizability and consensus of positive homogeneous multi-agent dynamical systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1936–1941, 2014.

[51] D. V. Dimarogonas and K. H. Johansson, "Stability analysis for multi-agent systems using the incidence matrix: quantized communication and formation control," *Automatica*, vol. 46, no. 4, pp. 695–700, 2010.

[52] S. Tonetti and R. M. Murray, "Stability and performance of non-homogeneous multi-agent systems on a graph," in *Proceedings of the IFAC World Congress*, 2010, http://www.cds.caltech.edu/~murray/preprints/tm11-ifac_s.pdf.

[53] W. Lenz, "Beiträge zum Verständnis der magnetischen Eigenschaften in festen Körpern," *Physikalische Zeitschrift*, vol. 21, pp. 613–615, 1920.

[54] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik*, vol. 31, no. 1, pp. 253–258, 1925.

[55] P. Erdős and A. Rényi, "On the evolution of random graphs," *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.

[56] S. Wolfram, "Universality and complexity in cellular automata," *Physica D: Nonlinear Phenomena*, vol. 10, no. 1-2, pp. 1–35, 1984.

[57] C. G. Langton, "Computation at the edge of chaos: phase transitions and emergent computation," *Physica D: Nonlinear Phenomena*, vol. 42, no. 1–3, pp. 12–37, 1990.

[58] F. Schweitzer and L. Behera, "Nonlinear voter models: the transition from invasion to coexistence," *The European Physical Journal B*, vol. 67, no. 3, pp. 301–318, 2009.

[59] J. Liu, V. Yadav, H. Sehgal, J. Olson, H. Liu, and N. Elia, "Phase transitions on fixed connected graphs and random graphs in the presence of noise," in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference*, pp. 734–739, IEEE, Seville, Spain, December 2005.

[60] F. Leon, "A multiagent system generating complex behaviours," in *Computational Collective Intelligence. Technologies and Applications: 5th International Conference, ICCCI 2013, Craiova, Romania, September 11–13, 2013, Proceedings*, C. Badică, N. T. Nguyen, and M. Brezovan, Eds., vol. 8083 of *Lecture Notes in Computer Science*, pp. 154–164, Springer, Berlin, Germany, 2013.

[61] F. Leon, "Design and evaluation of a multiagent interaction protocol generating behaviours with different levels of complexity," *Neurocomputing*, vol. 146, pp. 173–186, 2014.

[62] F. Leon, "Multiagent role allocation based on cognitive dissonance theory," *International Review on Computers and Software*, vol. 6, no. 5, pp. 715–724, 2011.

[63] F. Leon, "Self-organization of roles based on multilateral negotiation for task allocation," in *Multiagent System Technologies*, F. Klügl and S. Ossowski, Eds., vol. 6973 of *Lecture Notes in Computer Science*, pp. 173–180, Springer, Berlin, Germany, 2011.

[64] J. Nash, "Equilibrium points in n -person games," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 36, pp. 48–49, 1950.

[65] A. Bracciali, P. Mancarella, K. Stathis, and F. Toni, "Engineering stable multi-agent systems," in *Engineering Societies in the Agents World V: 5th International Workshop, ESAW 2004, Toulouse, France, October 20–22, 2004, Revised Selected and Invited Papers*, M. P. Gleizes, A. Omicini, and F. Zambonelli, Eds., vol. 3451 of *Lecture Notes in Computer Science*, pp. 322–334, Springer, Berlin, Germany, 2005.

[66] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations*, vol. 11, no. 1, 2009.

[67] R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, Calif, USA, 1993.

[68] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[69] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.

[70] V. G. Ivancevic and T. T. Ivancevic, *Complex Nonlinearity: Chaos, Phase Transitions, Topology Change and Path Integrals*, Springer, Berlin, Germany, 2008.

[71] A. Szolnoki and M. Perc, "Correlation of positive and negative reciprocity fails to confer an evolutionary advantage: phase transitions to elementary strategies," *Physical Review X*, vol. 3, no. 4, Article ID e041021, 2013.

[72] A. Byrski, R. Schaefer, M. Smolka, and C. Cotta, "Asymptotic guarantee of success for multi-agent memetic systems," *Bulletin of the Polish Academy of Sciences: Technical Sciences*, vol. 61, no. 1, pp. 257–278, 2013.

[73] A. Peleteiro, J. C. Burguillo, and A. L. Bazzan, "Emerging cooperation in the spatial IPD with reinforcement learning and coalitions," in *Intelligent Decision Systems in Large-Scale Distributed Environments*, vol. 362 of *Studies in Computational Intelligence*, pp. 187–206, Springer, 2011.