THE INFLUENCE OF THE UPDATE DYNAMICS ON THE EVOLUTION OF COOPERATION

Carlos Grilo [1, 2], **Luís Correia** [2]

[1] *Dep. de Engenharia Informática, Escola Superior de Tecnologia e Gestão, Instituto Politécnico de Leiria Campus 2, Morro do Lena - Alto do Vieiro, 2411-901 Leiria, Portugal*

E-mail: grilo@estg.ipleiria.pt

[2] *LabMAg, Dep. de Informática, Faculdade de Ciências, Universidade de Lisboa Bloco C6 Piso III, Campo Grande, 1749 - 016 Lisboa, Portugal*

E-mail: luis.correia@di.fc.ul.pt

Received: 30/09/08
Revised: 19/05/09

Abstract

We investigate the influence of the update dynamics on the evolution of cooperation. Three of the most studied games in this area are used: Prisoner's Dilemma, Snowdrift and the Stag Hunt. Previous studies with the Prisoner's Dilemma game reported that less cooperators survive with the asynchronous version of the game than with the synchronous one. On the other side, studies with the Snowdrift game are not conclusive about this subject. Based on simulations with these three games, played on different types of networks and using different levels of noise in the choice of the next strategy to be adopted by the agents, we conclude that, in general, an asynchronous dynamics favors the evolution of cooperation. Results concerning the monotonicity of these models and their sensitivity to small changes in the synchrony rate are also reported. This work is a contribution to a better understanding of the conditions under which cooperation can emerge and how different parameters may influence this emergence.

Keywords: evolution of cooperation, spacial evolutionary games, update dynamics, asynchronism.

1. Introduction

In a dynamical system, the *update method* defines the order in which the elements of the system are updated. *Synchronous updating* has been the most commonly used policy: at each time step, all the elements of the system are updated at exactly the same time. This practice has been widely questioned, the argument being that perfect synchronism is absent from the real world even in systems where synchronization processes exist. Besides, it has been shown that the dynamics of the system and the patterns generated, for example, in cellular automata, can be significantly affected if an asynchronous updating is used.[1,2,3] The most common alternative to synchronous updating is *sequential updating*, which is a special case of asynchronism: at each time step, exactly one element is updated. In order to use this type of updating policy, it is necessary to define how to choose the next element to be updated. It has been shown that the order in which the elements are updated may also influence the dynamics and the final state of the system.[4]

Synchronous and sequential updating are, thus, seemingly opposite ways of modeling the update dynamics of a dynamical system. In this paper we argue that these policies are both equally artificial models and we investigate the consequences of using an update method that allows the exploration of intermediate levels of asynchronism. We apply our study to spatial evolutionary games which are particular examples of non-linear dynamical systems.

For biologists, it has been challenging to explain the existence of cooperation in nature since, from an evolutionary point of view, this type of behavior is apparently less advantageous than selfish ones.[5] This problem is also of central importance in social sciences[6] and especially on the development and maintenance of artificial societies,[7] where it is also relevant to study how cooperation can be promoted and sustained. Spatial evolutionary games are models that are used to study these phenomena. In these models, a structured population of agents interacts during several time steps through a given *game* which is used as a metaphor for the type of interaction that is being studied. The population is structured in the sense that each agent can only interact with its neighbors. The underlying structure that defines who interacts with whom is called the *interaction topology*. After each interaction session, some or all the agents, depending on the *update dynamics* used, have the possibility of changing their strategies. The *strategy update process* is modeled using a so called *transition rule* that emulates the fact that agents tend to adapt their behavior to the context in which they live by imitating the most successful agents they know. It can also be interpreted as the selection step of an evolutionary process in which the least successful strategies tend to be replaced by the most successful ones.

The discussion about the influence of the update dynamics on spatial evolutionary games started after the famous work by Nowak and May,[8] who showed that cooperation can emerge and be sustained when the Prisoner's Dilemma game is played on a regular 2-dimensional grid by agents which always imitate their most successful neighbor. These results were contested by Huberman and Glance,[9] who argued against the synchronous updating that was used. They also reported results of simulations where cooperation was no longer sustainable when a sequential updating was used. After this, Nowak *et al*[10] tested their model under several conditions, including synchronous and sequential updating and showed that cooperation can be maintained for many different conditions, including sequential updating. However, the results are presented through system snapshot images of the model's state, which render it difficult to measure the way they are affected by the modification from synchronous to sequential updating. More recently, Newth and Cornforth,[11] studied a similar scenario using various sequential update methods besides the synchronous one. The authors found that the synchronous updating scheme supports more cooperators than the sequential ones.

In our previous work with the Prisoner's Dilemma game, we found that asynchronous updating supports, in general, more cooperators than synchronous updating when the game is played both in *regular*, *small-world*, *random* and *scale-free* networks.[12,13] We used a transition rule (see Sec. 2.4), which allows us to tune the level of noise present in the strategy update process and found that the influence of the updating dynamics depends mainly on the noise level.[14] We consider that there is noise when an agent fails to imitate the strategy of its most successful neighbor. We confirmed the results of Newth and Cornforth, obtained for regular networks and the *best-neighbor* transition rule, but we also found that asynchronous updating is detrimental to cooperation for very small noise values only. That is, for the most part of the noise domain, asynchronous updating supports more cooperators than the synchronous policy. Also, as we go from regular to random networks, asynchronous updating becomes beneficial to cooperation even for very small noise values. We also showed that the final outcome of the model is basically the same whether a deterministic or a stochastic asynchronous method is used.

In this paper we extend the study of the influence of the update method on spatial evolutionary games. We present formal expressions for the statistical properties of the update method used. We also propose measures to analyze the results obtained with numerical simulations and we extend the analysis to two more games: the Stag Hunt game and the Snowdrift game. The Snowdrift game is mathematically equivalent to the Hawk-Dove and the Chicken games. The different names are just a consequence of the big variety of real situations for which it can be used as a metaphor. There are at least two good reasons to extend this study to games other than the Prisoner's Dilemma. The first one is that both the similarities and the differences in the results achieved with different games can help us in the formulation of a better picture of the problem, leading to a better understanding of the conditions that allow the evolution of cooperation. The second reason follows from the difficulty that field researchers frequently experience in the evaluation of the relative value of the payoffs involved in concrete real situations.[15] Different payoff relations may define different games and that reinforces the need to experiment with several games.

We do not know any study about the influence of the update method on the Stag Hunt game. In what concerns the Snowdrift game, Tomassini *et al*[16] found that sequential updating supports less cooperators than the synchronous policy when the *best-neighbor* transition rule is used and that the opposite happens when a *proportional* transition rule is used. However, these two transition rules are just two particular cases of the noise level present in the strategy update process, which prevents any conclusion about the general influence of the update dynamics in this game. In Sec. 3.3 we will see that this influence can be better understood if we systematically explore the noise level present in the agents' decision process.

The paper is structured as follows: in Sec. 2 we describe the model we used in the simulations and in Sec. 3 we present and discuss the results. Finally, in Sec. 4 some conclusions are drawn and future work is advanced.

2. The Model

2.1. The games

In this section we describe the three games used in this study: the Prisoner's Dilemma (PD), the Snowdrift (SD) and the Stag Hunt (SH) games.[17] In all the three games, players can only cooperate (C) or defect (D). The payoffs are the following: R to each player if they both play C; P to each if they both play D; T and S if one plays D and the other C, respectively. These games differ in the relations existing between the payoff values: while in the PD game these values must obey $T > R > P > S$, in the SD game they must obey $T > R > S > P$ and in the SH game they must be such that $R > T \geqslant P > S$. Given these conditions, it follows that, in the PD game, D is the best action to take regardless of the opponent's decision. In the SD and SH games the best action depends on the opponent's decision but, while in the SD game the best thing to do is to take the opposite action the opponent takes, in the SH game the best is to take the same action as the opponent. For practical reasons, it is common to rescale the payoffs such that the games can be described by one parameter only.[18] The PD's payoffs are defined as $R = 1$, $T = b > 1$ and $S = P = 0$, where b represents the advantage of D players over C ones when they play the game with each other. In the SD game, payoffs are defined as $T = \beta \geqslant 1$, $R = \beta - 1/2$, $S = \beta - 1$ and $P = 0$ which leads to a cost-to-benefit ratio of mutual cooperation $r = 1/(2\beta - 1), 0 \leqslant r \leqslant 1$. Finally, the payoffs for the SH game are defined as $R = 1$, $T = P = h, 0 \leqslant h \leqslant 1$ and $S = 0$. Tables 1-3 show the payoff matrixes for the three games. On each cell, the first and second values are the payoff of the row player and the column player, respectively.

Table 1. Payoff matrix of the Prisoner's Dilemma game.

	C	D
C	$1, 1$	$0, b$
D	$b, 0$	$0, 0$

Table 2. Payoff matrix of the Snowdrift game.

	C	D
C	$\beta - \frac{1}{2}, \beta - \frac{1}{2}$	$\beta - 1, \beta$
D	$\beta, \beta - 1$	$0, 0$

Table 3. Payoff matrix of the Stag Hunt game.

	C	D
C	$1, 1$	$0, h$
D	$h, 0$	h, h

2.2. The interaction topologies

We used two types of interaction topologies: *small-world networks* (SWNs)[19] and *scale-free networks* (SFNs).[20] In order to build SWNs, first a toroidal regular 2D grid is built so that each node is linked to its 8 surrounding neighbors by undirected links; then, with probability ϕ, each link is replaced by another one linking two randomly selected nodes. We do not allow self or repeated links nor disconnected graphs. Networks built this way have the property that, even for very small ϕ values, the average path length is much smaller than in a regular network, maintaining a high clustering coefficient. Both these properties are very commonly observed in real social systems. As $\phi \to 1$, we get random networks with both small average path lengths and clustering coefficients.

SFNs have a power law degree distribution $P(k) \sim k^{-\gamma}$ that is also very common in real social networks. SFNs are built in the following way: the network is initialized with m fully connected nodes. Then, nodes are added, one at a time, until the network has the desired size. Each added node is linked to m already existing nodes so that the probability of creating a link with some existing node i is equal to $\frac{k_i}{\sum_j k_j}$, where k_i is the degree of i, that is, the number of nodes to which it is connected. This method of link creation is called *preferential attachment*, since the more links a node has, the greater is the probability of creating links to it. This has the effect that a small proportion of nodes has a big connectivity while the larger part has a very low connectivity.

2.3. The update dynamics

On each time step, agents first play a one round game with all their neighbors. Agents are pure strategists which can only play C or D. After this interaction stage, each agent updates its strategy with probability α using a *transition rule* (see next section). The update is done synchronously by all the agents selected to engage in this revision process. The α parameter is called the *synchrony rate* and is the same for all agents. This type of update mehtod is called *asynchronous stochastic dynamics* (ASD).[21] It allows us to cover all the space between synchronous and sequential updating. When $\alpha = 1$ we have a synchronous model, where all the agents update at the same time. As $\alpha \to \frac{1}{n}$, where n is the population size, the model approaches sequential updating, where exactly one agent updates its strategy at each time step.

When ASD is used, the number of updating agents on each time step is binomially distributed. If we define α as $\frac{a}{n}$, where $a \leqslant n$, then the expected value of active agents on each time step is equal to $a = \alpha n$ with a corresponding variance of $a(1-\frac{a}{n}) = n\alpha(1-\alpha)$. The probability that an agent x is chosen k times in v single steps is also binomially distributed, i.e.

$$P(x \text{ is chosen } k \text{ times}) = \binom{v}{k}\left(\frac{a}{n}\right)^k\left(1-\frac{a}{n}\right)^{v-k}. \quad (1)$$

In Ref. 4 expressions are given for the expected value $E(X)$ and variance V(X) of the number X of single steps between two updates of the same agent under sequential dynamics ($\alpha = \frac{1}{n}$). Here, we generalize these expressions for the more general case of $\alpha = \frac{a}{n}$. As in the sequential case, the expressions for X are the same as for the number Z of single steps between an update of agent x and an update of a given neighbor of x:

$$E(X) = \sum_{k=1}^{\infty} i\frac{a}{n}\left(1-\frac{a}{n}\right)^{i-1} = \frac{n}{a} = \frac{1}{\alpha}, \quad (2)$$

$$V(X) = \sum_{k=1}^{\infty}[i-E(X)]^2\frac{a}{n}\left(1-\frac{a}{n}\right)^{i-1} = \frac{n}{a}(\frac{n}{a}-1) = \frac{1}{\alpha}(\frac{1}{\alpha}-1). \quad (3)$$

ASD models the fact that, at each moment, more than one agent, but not necessarily all of them, may update their strategy. Usually, asynchronism is understood as sequential updating. As an example, in all the works mentioned above, asynchronous dynamics means sequential updating. However, the real world seems to lie somewhere between synchronism and sequentiality and, so, both types of updating dynamics can be considered as artificial. In a population of interacting agents, many decision processes can occur at the same time but not necessarily involving all the agents. If these were instantaneous phenomena we could model the update dynamics of the system as if they occurred one after another but that is not usually the case. These processes can take some time, which means that their output is not available to other ongoing decision processes. Even if we consider them as being instantaneous, the time that information takes to be transmitted and perceived implies that their consequences are not immediately available to other agents. As we have seen above, ASD also models the fact that, at each time step, the number of agents updating their strategy is not always the same, which is a reasonable assumption. Apart from these arguments, as we will see in the following sections, the fact that the α parameter allows us to explore intermediate levels of asynchronism is also useful in the analysis of the influence of the update dynamics.

2.4. *The strategy update process*

The strategy update process is done using a transition rule that models the fact that agents tend to imitate the most successful agents they know. In our simulations, we used, the *generalized proportional* transition rule (GP)[10]. Let G_x be the average payoff earned by agent x, N_x be the set of neighbors of x and c_x be equal to 1 if x's strategy is C and 0 otherwise. According to this rule, the probability that an agent x adopts C as its next strategy is

$$p_C(x,K) = \frac{\sum_{i \in N_x \cup x} c_i (G_i)^{\frac{1}{K}}}{\sum_{i \in N_x \cup x} (G_i)^{\frac{1}{K}}}, \tag{4}$$

where $K \in]0,+\infty[$ is the noise present in the strategy update process. Noise is present in this process if there is some possibility that an agent imitates strategies other than the one used by its most successful neighbor. Small noise values favor the choice of the most successful neighbors' strategies. Also, as noise diminishes, the probability of imitating an agent with a lower payoff becomes smaller. When $K \rightarrow 0$ we have a deterministic *best-neighbor* rule such that i always adopts the best neighbor's strategy. When $K = 1$ we have a simple *proportional* update rule. Finally, for $K \rightarrow +\infty$ we have random drift where payoffs play no role in the decision process. For the moment, our analysis considers only the interval $K \in]0,1]$. In this interval the decision process is strongly guided by the payoffs earned by the agents.

2.5. *Simulation setup*

All the simulations were done with populations of $50 \times 50 = 2500$ agents, randomly initialized with 50% of Cs and 50% of Ds. When the system is running synchronously, i.e., when $\alpha = 1$, we let it run during a transient period of 900 iterations. After this, we let the system run during 100 more iterations and, at the end, we take as output the average proportion of cooperators during this period. When $\alpha \neq 1$, the number of selected agents at each time step may not be equal to the size of the population and it may vary between two consecutive time steps. In order to guarantee that these runs are equivalent to the synchronous ones concerning to the total number of individual updates, we let the system first run until 900×2500 individual updates have been done. After this, we sample the proportion of cooperators during 100×2500 individual updates and we average it by the number of time steps needed to do these updates. Each simulation is a combination of a specific game (parameters b, r or h for the PD, SD and SH games, respectively), an interaction topology (ϕ or m, for SWNs and SFNs, respectively), a noise value K for the transition rule and a α value defining the synchrony rate of the model. All the possible combinations of the values shown in Table 4 were tested. For each combination, 30 runs were made and the average of these runs is taken as the output.

For each combination of the ϕ/m and K parameters, where ϕ/m means that ϕ or m should be used depending on the type of interaction topology, we produced a chart like the one of Fig. 1. Each line in the chart is a combination of the ϕ/m, K and $b/r/h$ parameters and, for simplicity, for the rest of the paper we will refer to this type of combination simply as a *line*. Since we tested with 8 K values, 5 ϕ values, 3 m values and 11 $b/r/h$ values, this gives $8 \times 5 + 8 \times 3 = 64$ of these charts and $64 \times 11 = 704$

lines per game. The statistical measures we used for the analysis in the next section are based on *lines* and charts as the one of Fig 1.

Table 4. Parameter values used in the simulations.

Parameter	Values
b (PD)	$1, 1.1, 1.2, ..., 1.8, 1.9, 2$
r (SD)	$0, 0.1, 0.2, ..., 0.8, 0.9, 1$
h (SH)	$0.4, 0.45, 0.5, ..., 0.8, 0.85, 0.9$
ϕ (SWNs)	$0, 0.01, 0.05, 0.1, 1$
m (SFNs)	$2, 4, 8$
α (ASD)	$0.1, 0.2, 0.3, ..., 0.8, 0.9, 1$
K (noise)	0, 1/100, 1/10, 1/8, 1/6, 1/4, 1/2, 1

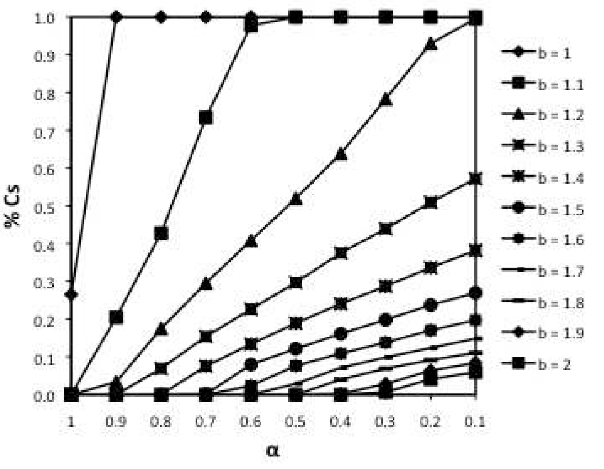

Fig. 1. Proportion of Cs for the PD game on SFNs with $m = 2$ and with $K = 1$.

3. Results

We analyze three aspects of the results: the *local sensitivity* of the model to α, which is a measure of how much the proportion of C players changes when α is changed by a small value; *monotonicity*, where we verify if the proportion of cooperators changes always in the same direction as we change α; the impact that the update dynamics has on the proportion of cooperators that survive, where we answer the question "Is asynchronism beneficial or detrimental to the evolution of cooperation?".

3.1. Local sensitivity

We compute two different quantities, *maxStep* and $step(0.9,1)$, in order to estimate the local sensitivity of the model to α.[21] Taking Fig. 1 as an example, for each *line*, we make

$$maxStep = max(|C(\alpha) - C(\alpha - 0.1)|), \quad \alpha = 1, 0.9, ..., 0.2 \tag{5}$$

and

$$step(1, 0.9) = |C(1) - C(0.9)|, \tag{6}$$

where $C(\alpha)$ represents the proportion of cooperators achieved with the given α value. The *maxStep* quantity is an estimatiion of the model's sensitivity along all the α domain, while $step(1,0.9)$ tells us how sensitive the model is when we change from perfect synchronism to a near synchronous updating.

Figs. 2, 3 and 4 show the sensitivity values, respectively for the PD, SD and SH games, for all the possible parameter combinations (*lines*) that were simulated. For the three games, the most part of the points are below 0.1, for both the *maxStep* and $step(0.9,1)$ coordinates. However, we can see that, specially in the PD and the SH games, there are some situations of big sensitivity. While in the PD game these situations happen mainly for SFNs (Fig. 1 has some examples of this), in the SD and SH games there is no clear pattern about the influence of the interaction topology on the local sensitivity. We can also see that in the PD and SD games, the *maxStep* often has the same value as $step(0.9,1)$ (values in the chart's diagonal). This happens 23.2% and 52.8% of the times, respectively for the PD and SD games. It means that the model is sensitive mainly when we change from a synchronous to a near synchronous regime. This result is specially relevant because it suggests that if we are modeling a real situation where synchronization processes are at play, we should not rely on results achieved under perfect synchronism only and that we should inspect the behavior of the model under a quasi-synchronous regime.

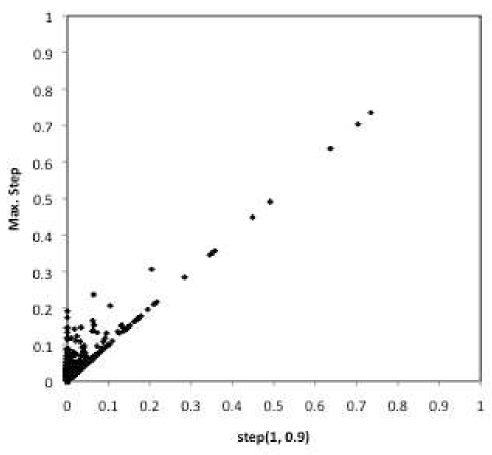

Fig. 2. *maxStep* and $step(1, 0.9)$ values for the PD game.

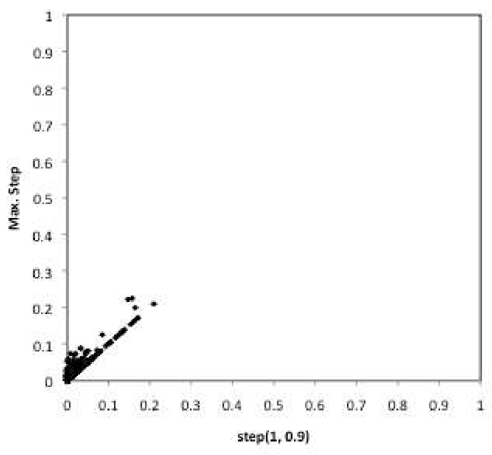

Fig. 3. *maxStep* and $step(1, 0.9)$ values for the SD game.

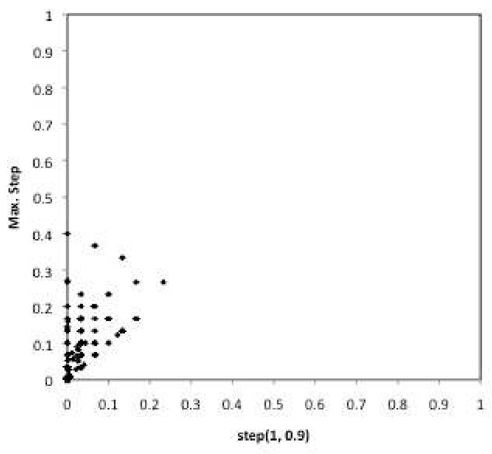

Fig. 4. *maxStep* and $step(1, 0.9)$ values for the SH game.

Finally, a word to say that, despite the fact that the most part of the steps are below 0.1, there are plenty of situations where the difference between the level of cooperation achieved with $\alpha = 1$ and $\alpha = 0.1$ is very significant. Again, Fig. 1 illustrates this very well.

3.2. Monotonicity

The exploration of intermediate levels of asynchronism allows us to verify if the level of cooperation changes monotonically as we change α. In order to assess the monotonicity of the model, for each *line*, we compute the quantity

$$M = \left(\sum_{i=2}^{10} |C(0.1i) - C(0.1(i-1))|\right) - |C(1) - C(0.1)|, \tag{7}$$

where, again, $C(\alpha)$ is the level of cooperation achieved with a given α value. $M = 0$ if a *line* is monotonic. If a *line* is non-monotonic, the more and larger fluctuations there are, the larger will M be. Table 5 shows the average M values per phi/m and K parameter combination for the SD game, which is the one presenting the larger M values. We can see that the most part of the values are equal or very close to 0. This means that, essentially, the proportion of cooperators changes monotonically with α in the three games. This is an important result since it means that, in general, the maximum influence the update method may have in the evolution of cooperation can be assessed by inspecting the output achieved with α values near the extremes of the α domain.

Table 5. Average M values for the SD game.

		ϕ					m		
		0	0.01	0.05	0.1	1	2	4	8
K	0	0.14	0.16	0.07	0.03	0.01	0.00	0.00	0.03
	1/100	0.11	0.08	0.05	0.04	0.00	0.00	0.00	0.04
	1/10	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.01
	1/8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	1/6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	1/4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	1/2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

3.3. *Is anyone favored by asynchronism?*

Is asynchronism beneficial or detrimental to the evolution of cooperation? In order to answer this question, we first compute, for each *line*, the quantity

$$D = C(0.1) - C(1), \qquad (8)$$

which is the difference of the proportion of cooperators achieved with $\alpha = 0.1$ and $\alpha = 1$. Positive or negative D values mean, respectively, that asynchronism supports more or less cooperators than synchronism. It is possible to base the analysis on this difference due to the result reported in the previous section concerning the monotonic behavior of the model. Tables 6, 7 and 8 show, respectively for the PD, SD and SH games, the average D value for each ϕ/m and K combination (we remember that each such combination corresponds to a chart such as the one of Fig. 1). We complement the D average values with bold numbers in situations where at least one *line* with a negative D value exists.

Table 6. Influence of the update method in the PD game.

		ϕ					m		
		0	0.01	0.05	0.1	1	2	4	8
K	0	**-0.04**	**-0.03**	**-0.01**	**0.03**	**0.15**	**0.14**	**0.07**	**0.07**
	1/100	**-0.02**	**-0.02**	**-0.01**	**0.03**	**0.18**	0.13	0.08	0.09
	1/10	**-0.01**	0.02	0.04	0.06	0.14	0.06	0.08	0.09
	1/8	0.03	0.03	0.06	0.07	0.14	0.07	0.08	0.09
	1/6	0.05	0.06	0.07	0.08	0.13	0.09	0.09	0.08
	1/4	0.07	0.08	0.10	0.11	0.14	0.12	0.12	0.09
	1/2	0.12	0.12	0.13	0.15	0.19	0.22	0.19	0.14
	1	0.14	0.16	0.18	0.19	0.27	0.41	0.30	0.19

Table 7. Influence of the update method in the SD game.

		ϕ					m		
		0	0.01	0.05	0.1	1	2	4	8
K	0	**-0.09**	**-0.07**	**-0.04**	**0.01**	**0.09**	**0.08**	**0.08**	**0.07**
	1/100	**-0.09**	**-0.08**	**-0.03**	**0.02**	0.11	0.08	0.08	0.06
	1/10	**0.01**	**0.01**	0.03	0.04	0.10	0.10	0.08	0.04
	1/8	0.02	0.02	0.04	0.05	0.10	0.10	0.08	0.05
	1/6	0.04	0.04	0.05	0.05	0.10	0.11	0.08	0.05
	1/4	0.05	0.05	0.06	0.06	0.10	0.12	0.09	0.05
	1/2	0.07	0.07	0.07	0.08	0.10	0.15	0.09	0.05
	1	0.08	0.09	0.09	0.09	0.10	0.17	0.10	0.06

Table 8. Influence of the update method in the SH game.

		ϕ					m		
		0	0.01	0.05	0.1	1	2	4	8
K	0	0.01	0.01	0.01	0.01	0.03	0.05	0.01	0.03
	1/100	0.02	0.04	0.01	0.01	0.02	0.03	0.01	0.02
	1/10	0.03	0.01	0.00	0.02	0.03	0.01	0.04	0.02
	1/8	0.01	0.01	0.05	0.00	0.02	0.04	0.02	0.02
	1/6	0.05	0.01	0.00	0.06	0.03	0.01	0.06	0.03
	1/4	0.06	0.08	0.00	0.02	0.04	0.07	0.05	0.02
	1/2	0.08	0.05	0.03	0.06	0.05	0.08	0.06	0.03
	1	0.08	0.04	0.04	0.06	0.08	0.12	0.06	0.04

Table 6 shows that negative D values exist for very low noise values. This is coherent with the results obtained in the above mentioned work by Newth and Cornforth with the PD game. However, the table also shows that D is positive for the most part of the noise domain and that it grows with noise. This completely contradicts the established idea that asynchronous updating supports less cooperators than the synchronous one in this game. Table 7 shows that the results for the SD game are very similar to the ones obtained with the PD game, however, with smaller absolute D values. This is also coherent with the results obtained by Tomassini *et al* (we recall that the best-neighbor and the proportional transition rules used by these authors are equivalent to the generalized proportional transition rule, respectively, with $K = 0$ and $K = 1$). But, also in this case, exploring several noise values allows us to have a much better idea about how the update method influences the evolution of cooperation. Finally, Table 8 shows that in the SH game an asynchronous update always supports more cooperators than the synchronous one.

We may, thus, conclude that, in general, asynchronism is beneficial to the evolution of cooperation. This conclusion is reinforced by the fact that asynchronism is detrimental for very unrealistic situations only, where noise levels are very low. Indeed, another common feature to the three games is the fact that, excepting some small fluctuations, the benefit of asynchronism grows with noise.

There is, however, one significant difference in how asynchronism affects the evolution of cooperation in the three games which is not possible to identify only by inspecting tables 6-8. Thus, as Figs. 1

and 5 illustrate, in the PD and SD games there are several *lines* per chart for which cooperators and defectors coexist. This is not the case with the SH game, where Cs and Ds coexist only for a very small number of the tested h values. For example, in the chart of Fig. 6, Cs and Ds only coexist for $h = 0.65$. Below this value only Cs survive and above it only Ds survive. This is because for, the most part of the h values, the game is too easy or to difficult for C players, precluding any influence of the α parameter. The few h values for which Cs and Ds survive vary considerably (from 0.4 to 0.9) with the noise level and the type of interaction topology, and this is why we explore this interval instead of concentrating on a smaller h interval. Of course that, given a specific ϕ/m and K combination, if we concentrate on a smaller interval around h values where Cs and Ds coexist, we will get a larger average D value. Just to confirm this, we explored the h parameter around 0.65, from 0.6 to 0.7 by steps of 0.01 for the situation illustrated in Fig. 6. In this experiment, Cs and Ds coexist for h values from 0.63 to 0.67 and the average D value is 0.15, which is the double of the average D value of Fig. 6. In what concerns the interpretation of the values of Table 8 (SH game), this means that we should be aware that, in situations where Cs and Ds coexist, the real influence of asynchronism is substantially larger than the values shown in Table 8. It also means, that for similar average D values, as in Figs. 5 and 6, the influence in situations where Cs and Ds coexist is, in general, larger for the SH game than for the PD and specially the SD games.

The influence of asynchronism on the evolution of cooperation is also affected by the interaction topology, for the PD and the SD games. For these two games, the benefit of asynchronism grows as ϕ increases, on SWNs, and as m decreases, on SFNs (see tables 6 and 7). Although we do not know yet what are the mechanisms that lead to this pattern of behavior and how they operate, we hypothesize that it is related to the clustering coefficient of the interaction topology, which decreases for the two types of network as ϕ and m, respectively, increase and decrease. Another possibility would be the average path length between the nodes. However, while the average path length decreases on SWNs as we increase the ϕ value, it decreases on SFNs as we decrease the value of m. More work must be done, however, in order to confirm this, and to verify why this pattern of behavior is absent from the SH game as it seems to be.

Finally, we would like to stress that using an update method able to cover all the space between synchronous and near sequential updating, allows a deeper analysis of the model being studied. It allows, for example, the identification of existing phase transitions in the level of cooperation achieved: often the proportion of Cs is 0 for $\alpha = 1$ and it remains there until a given $\alpha = c$ value is reached. Then, suddenly, the level of cooperation starts to increase, sometimes in a significant way, as α decreases from c to 0.1. For many of such situations c is very close or equal to 1 (see, for example, Fig. 1). This may suggest that in these cases the existence of some degree of cooperation is the most probable outcome in the system being modeled since it exists for almost the entire α domain. This type of analysis is not possible if only the extremes of the update dynamics' space are explored, even in situations where the model answers monotonically to changes in the synchrony rate.

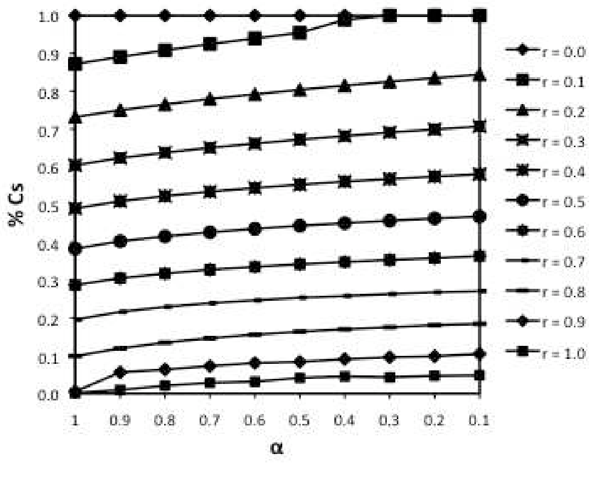

Fig. 5. Proportion of Cs for the SD game on SFNs with $m = 4$ and with $K = 1/8$. The average D value is 0.082.

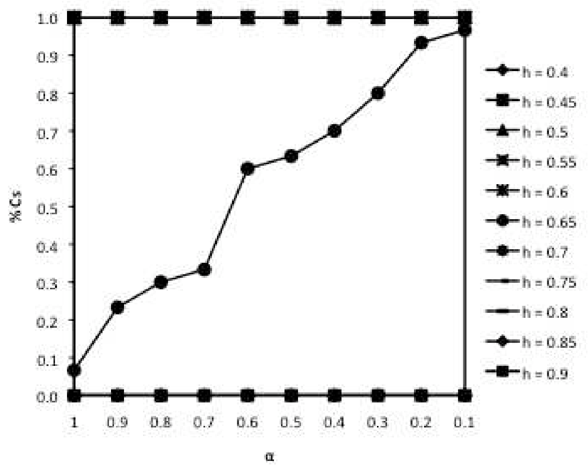

Fig. 6. Proportion of Cs for the SH game on SWNs with $\phi = 0.01$ and with $K = 1/4$. Cs and Ds only coexist for $h = 0.65$. The average D value is 0.082.

4. Conclusions and Future Work

The real world seems to lie somewhere between perfect synchronism and sequential dynamics. However, it is common practice to use only one of these two types of update dynamics, rarely the two, in the modeling and study of dynamical systems. Synchronism, for example, is often used even in cases where there is no evidence that synchronization processes exist. In this paper we argued that intermediate levels of asynchronism should be used in the study of dynamical systems. We studied the influence of the update method in spatial evolutionary games, which are special cases of non-linear dynamical systems used to study the evolution of cooperation in many areas like, for example, biology, sociology, economics and artificial societies.

In this work we used three of the most studied games, the Prisoner's Dilemma, the Snowdrift and the Stag-Hunt games, and the *asynchronous stochastic dynamics* update method, which allows the exploration of all the space between synchronous and sequential updating. This update method allows the study of some aspects as, for example, the local sensitivity and the monotonicity of the system, that are not possible to study if only synchronous and sequential updating are used. We found that, in general, the three games are not very sensitive to small changes in the synchrony rate. However, in a significative number of situations, the games are sensitive mainly when a change from a synchronous to a near synchronous regime is made. This, means that special care must be taken when studying real systems where synchronization processes exist. The results also show that, for the most part of the tested situations, the model responds monotonically to changes in the synchrony rate. This behaviour is usually taken for granted. But, the results reported in recent works on elementary cellular automata systems,[21] with which spatial evolutionary games have many resemblances, show that the behaviour of these systems may be very different on intermediate values of the synchrony rate.

We also found that, in general, an asynchronous discipline supports more cooperators than a synchronous one. Previous studies with the Prisoner's Dilemma game reported that asynchronism is detrimental to the evolution of cooperation when agents use the *best-neighbor* transition rule. On the other hand, the work by Tomassini *et al*, with the Snowdrift game, is not conclusive about this subject: asynchronous updating is detrimental to the evolution of cooperation if the *best-neighbor* rule is used, and it is beneficial if the *proportional* rule is used. The generalized transition rule we applied in our work allows these results to be understood in a broader context: they are mainly a consequence of the noise present in the strategy update process. This rule allows us to tune the noise level, K, and is able to emulate both the *best-neighbor* rule ($K = 0$) and the *proportional* rule ($K = 1$). We found that asynchronism is detrimental to the evolution of cooperation in very unrealistic situations only, when noise is absent from the decision process or when it is very small. The results also show that, the more noise is present in the strategy update process, the more cooperators are favored by asynchronism. Finally, for the most part of the tested conditions, asynchronism supports more cooperators than synchronism. The confidence in the general character of this result is reinforced by the fact that it applies to the three studied games and to various interaction topologies.

One of the future extensions to this work will explore other types of transition rules. The rule we used here emulates a complete neighborhood monitoring where each agent takes into account the payoffs of all its neighbors in the decision process. But, a complete neighborhood monitoring is not always possible and so we want to verify what happens in this case. The *Sigmoid* [17] transition rule is a good candidate since it also allows us to tune the noise level present in the decision process. Another direction for this work will consider dynamic interaction topologies[22] where agents are allowed to establish new connections with other agents and break existing ones. We hope that the results obtained under different conditions lead us to a deeper insight about the influence of asynchronism on the evolution of cooperation.

Acknowledgments

This work was partially supported by FCT/MCTES grant No. SFRH/BD/37650/2007.

5. References

1. T. E. Ingerson and R. L. Buvel, "Structure in asynchronous cellular automata", Phys. D, **10**, 59-68 (1984).
2. H. Bersini and V. Detours, "Asynchrony induces stability in cellular automata based model", *Proceedings of the Artificial Live IV Conference*, 382-387 (1994).
3. E. D. Lumer and G. Nicolis, "Synchronous versus asynchronous dynamics in spatially distributed systems", Phys. D, **71**, 440–452 (1994).
4. B. Schönfich and A. de Roos, "Synchronous and asynchronous updating in cellular automata", *BioSystems*, **51(3)**, 123-143 (1999).
5. J. M. Smith, "Evolution and the theory of games", Cambridge University Press (1982).
6. R. Axelrod, "The evolution of cooperation", Penguin Books (1984).
7. J. C. Oh, "Cooperating search agents explore more than defecting search agents in the internet information access",*Proceedings of the 2001 Congress on Evolutionary Computation, CEC2001*, 1261-1268 (2001).
8. M. Nowak and R. M. May, "Evolutionary games and spatial chaos", *Nature*, **359**, 826–829 (1992).
9. B. Huberman and N. Glance, "Evolutionary games and computer simulations", *Proceedings of the National Academy of Sciences*, **90**, 7716-7718 (1993).
10. M. Nowak, S. Bonhoeffer and R. M. May, "More spatial games", *International Journal of Bifurcation and Chaos*, **4(1)**, 33-56 (1994).
11. D. Newth and D. Cornforth, "Asynchronous spatial evolutionary games: spatial patterns, diversity and chaos", *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, 3463-2470 (2007).
12. C. Grilo and L. Correia, "Asynchronous stochastic dynamics and the spatial prisoner's dilemma game", *Proceedings of the 13th Portuguese Conference on Artificial Intelligence, EPIA 2007*, 235-246 (2007).
13. C. Grilo and L. Correia, "The influence of asynchronous dynamics in the spatial prisoner's dilemma game", *Animals to Animats - 10th International Conference on the Simulation of Behavior (SAB'08)*, 362-371 (2008).
14. C. Grilo and L. Correia, "What makes the spatial prisoner's dilemma game sensitive to asynchronism?", *Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems, Alife XI*, 212-219 (2008).
15. C. Hauert and M. Doebeli, "Spatial structure often inhibits the evolution of cooperation in the snowdrift game", *Nature*, **428**, 643–646 (2004).
16. M. Tomassini and L. Luthi and M. Giacobini, "Hawks and Doves on small-world networks", *Phys. Rev. E*, **73(1)**, 016132 (2006).
17. G. Szabó, "Evolutionary games on graphs", *Phys. Rep.*, **446**, 97-216 (2007).
18. J. M. Pacheco and F. C. Santos, "Network dependence of the dilemmas of cooperation", *Science of Complex Networks: From Biology to the Internet and WWW, CNET 2004*, **776**, 90-100 (2005).
19. D. Watts and S. H. Strogatz, "Collective dynamics of small-world networks", *Nature*, **393**, 440-442 (1998).
20. A. Barabasi and R. Albert, "Emergence of scaling in random networks", *Science*, **286**, 509-512 (1999).
21. N. Fatès and M. Morvan, "An experimental study of robustness to asynchronism for elementary cellular automata", *Complex Systems*, **16(1)**, 1-27 (2005).
22. L. Luthi, M. Giacobini and M. Tomassini, "Synchronous and Asynchronous Network Evolution in a Population of Stubborn Prisoners", *IEEE Symposium on Computational Intelligence and Games*, 225-232 (2005).