A Method to Derive Local Interaction Strategies for Improving Cooperation in Self-Organizing Systems*

Christopher Auer, Patrick Wüchner, and Hermann de Meer

Faculty of Informatics and Mathematics,
University of Passau, Innstraße 43,
94032 Passau, Germany
{auerc,wuechner,demeer}@fim.uni-passau.de

Abstract. To achieve a preferred global behavior of self-organizing systems, suitable local interaction strategies have to be found. In general, this is a non-trivial task. In this paper, a general method is proposed that allows to systematically derive local interaction strategies by specifying the preferred global behavior. In addition, the resulting strategies can be evaluated using Markovian analysis. Then, by applying the proposed method exemplarily to the iterated prisoner's dilemma, we are able to systematically generate a cooperation-fostering strategy which can be shown to behave similar to the "tit for tat with forgiveness" strategy that, under certain circumstances, outperforms the well-known "tit for tat" strategy used, for instance, in BitTorrent peer-to-peer file-sharing networks.

1 Introduction

In self-organizing systems (SOSs), numerous entities interact with their neighborhood following certain interaction strategies. The system's entities and their strategies define the *micro-level* of the SOS. From the behavior on the micro-level, a global behavior of the SOS emerges at the *macro-level*. In general, the micro-level entities of an SOS rely on local knowledge only, i.e., solely on the information that is provided by the entities within their vicinity. Therefore, we call these entities and their strategy *vicinity-dependent*. For instance, to find the shortest path from the ant colony to a food place, an ant follows pheromone trails laid out by other ants in its vicinity (cf. [1]). Hence, the behavior of the ants is vicinity-dependent and results in an emergent macro-level behavior, i.e., the discovery of shortest paths.

When engineering and designing SOSs, the strategies on the micro-level need to be specified to attain the preferred emergent behavior on the macro-level. Since the system's entities are purely vicinity-dependent, this is in general a

* This research is partially supported by the AutoI project (STREP, FP7 Call1, ICT-2007-1-216404) and by the Networks of Excellence EuroFGI (IST 028022) and EuroNF (IST 216366).

non-trivial task. The main contribution of this paper is a generic method to derive suitable micro-level strategies.

Our solution approach is based on the following observation: The task of specifying suitable strategies would be less challenging if an entity and its strategy could rely on global knowledge, i.e., information of the SOS's global state and/or the other entities' strategies. In the following, we call an entity (and its strategy) that relies on global knowledge *omniscient.*

As an example, in this paper, we investigate a class of systems where cooperation is of particular importance, most prominently for peer-to-peer (P2P) file-sharing systems like BitTorrent (see [2]), where the dissemination of content relies on the collaboration of numerous entities, i.e., the peers. Often, game-theoretic approaches are used to model the peers in such systems, where a common game-theoretic model is the iterated prisoner's dilemma (IPD; cf. [3–7]). The IPD assumes that the peers are rationally thinking players that try to maximize their individual benefit and that a player's interaction strategy is vicinity-dependent, i.e., each player can only observe the behavior of the other players but does not know the other players' strategies. For such systems, a vicinity-dependent interaction strategy is sought that fosters global cooperation on macro-level. Based on the assumption that the strategy does not know its opponents' strategy, this is a challenging task in general. Often, it would be easier to find an omniscient strategy which knows the other players' strategies, and hence, could optimize its behavior to foster the preferred macro-level behavior.

In our method, we use an artificial omniscient entity called the Laplace's Demon (LD): This name stems from a hypothetical demon envisioned by the French mathematician and astronomer Pierre-Simon Laplace (1749–1827). This demon exactly knows the state of the whole universe and, hence, is able to predict future states by Newton's laws of physics.[1] The behavior of our LD is then investigated by a time-series analyzing algorithm to derive a purely vicinity-dependent strategy, which, in contrast to an omniscient strategy, can be effectively implemented into technical SOSs.

The method is generic in the sense that it does not focus on a specific application area or modeling formalism. The most restrictive constraint is the requirement that the system's model can be evaluated using discrete-event simulation. However, this requirement does not impose a hard constraint on a large class of engineerable systems.

Our approach allows to explicitly foster the preferred macro-level behavior and also provides means to analyze to which extent the preferred macro-level behavior is reached.

The remainder of this paper is structured as follows: In Sec. 2, we discuss related work. Then, the IPD as a model for cooperation in SOSs is shown in Sec. 3. Section 4 briefly introduces a time-series analyzing algorithm — the CSSR algorithm developed by Shalizi et al. (cf. [8, 9]). In Sec. 5, we present in more detail our generic core method that is based on LDs. We apply this method in

[1] Pierre-Simon Laplace formulated this idea in the foreword to his essay "Essai philosophique sur les probabilités" from 1814.

Sec. 6 to derive vicinity-dependent interaction strategies for IPD-like problems. Finally, Sec. 7 draws conclusive remarks and gives an outlook to future work.

2 Related Work

We distinguish two complementary approaches for deriving local interaction strategies of entities in SOSs: At one extreme, it is assumed that the entities explicitly disseminate and periodically update global state information within the system, i.e., the entities are omniscient. At the other extreme, the system entities remain purely vicinity-dependent.

Omniscient entities are able to optimize their strategies to attain the preferred macro-level behavior based on global state information. Examples for such systems are distance vector and link state based algorithms for routing protocols as presented in [10, 11], where the omniscient entities are able to calculate optimal routing paths. However, the periodic dissemination and local storage of global state information clearly suffers from scalability issues in large systems (cf. [12]).

Many application-driven approaches to derive *vicinity-dependent* strategies exist. For example, the AntNet routing protocol to find shortest routing paths in mobile networks was developed by investigating the exchange of pheromones between ants (cf. [1]). In [13], the approach of epidemic information dissemination uses the principle behind the spread of a disease to disseminate information in distributed systems. To form topologies in P2P overlay networks, a vicinity-dependent strategy was developed in [14] by investigating cell adhesion processes. Further examples are intuitive approaches like the random walk search algorithm for P2P networks in [15] and the gossip-based routing algorithm for ad hoc networks in [16], where messages are forwarded to randomly chosen neighbors on a best effort basis. Unfortunately, these application-driven approaches do not provide a generic method that can be used in a wider range of application areas. Hence, finding purely *vicinity-dependent* strategies remains a non-trivial task in general, and, to the best of the authors' knowledge, no general method yet exists.

3 The Iterated Prisoner's Dilemma

We now give a primer to the IPD and introduce the problem to be solved with the proposed method in Sec. 6.

In SOSs, where the collaboration between the entities is of particular importance, micro-level interaction strategies are sought from which global cooperation on the system's macro-level emerges. For instance, to disseminate content, a P2P file-sharing network needs to define strategies that foster global cooperation between all peers, i.e., each peer shares its bandwidth and content to other peers. The IPD is a commonly used modeling formalism to study cooperation in such SOSs (see, e.g., [3–5]).

The IPD models the interaction at discrete time steps between two entities called the players. At each time step of the IPD, each of the two players can choose either to cooperate (C) or to defect (D). Both players make their current choice independently from each other but the choice may well depend on the outcome of previous iterations. Both players present their choice and get a payoff, e.g., according to the following utility matrix **U**, which is taken from [17]:

$$\mathbf{U} = \begin{array}{c c} & \begin{array}{cc} \text{D} & \text{C} \end{array} \\ \begin{array}{c} \text{D} \\ \text{C} \end{array} & \begin{bmatrix} (1,1) & (4,0) \\ (0,4) & (3,3) \end{bmatrix} \end{array} . \tag{1}$$

The rows of **U** correspond to the choice of the first player, the columns to the choice of the second player, and the entries of **U** correspond to the payoffs, where the first element of each tuple is the payoff for the first player and the second element is the payoff for the second player.

It can easily be seen that cooperation on both sides maximizes the sum of the payoffs of both players, that is, the global welfare. However, cooperation on both sides may increase the temptation of one player to defect in order to maximize its personal payoff to a value of 4, whereas the payoff of the other player is minimized to a value of 0. This rational thinking process might lead both players to defect, which minimizes the global welfare and results in a lower individual payoff for both players compared to the case of mutual cooperation. In the IPD, the prisoner's dilemma is iterated and, hence, the outcome of past iterations can be incorporated into the decision process of the players.

When modeling the micro-level interactions of an SOS by the IPD, a strategy is sought from which global cooperation emerges on the systems' macro-level. Such a strategy should adjust its behavior to foster cooperation if its opponent is willing to cooperate and it should defect otherwise. The latter implicitly fosters global cooperation, since it makes defection less profitable than mutual cooperation. However, finding such a strategy is a non-trivial task in general since the IPD assumes that the players' strategies are purely vicinity-dependent, i.e., each player can incorporate only previous behavior of its opponent into its decision process but does not know its opponent's strategy.

In this paper, we systematically derive such a cooperation-fostering strategy by investigating an omniscient player, i.e., the LD. The LD knows its opponent's strategy and is, hence, able to easily adjust its behavior to foster cooperation. By investigating the behavior of the LD, a vicinity-dependent strategy for the IPD is derived. To do so, we first propose a generic method to derive vicinity-dependent interaction strategies for SOSs (see Sec. 5). The method is then applied to the IPD (see Sec. 6).

4 The CSSR Algorithm

This section briefly describes the Causal-State Splitting Reconstruction (CSSR) algorithm, which we utilize in our method as described later in Sec. 5. We choose this algorithm because it is sufficiently abstract to be applicable in a wide range of application scenarios. The interested reader is referred to [8, 9] for a comprehensive description of the algorithm.

The CSSR algorithm takes a sequence of $N \in \mathbb{N}$ symbols as input. This sequence is drawn from a discrete alphabet $\mathcal{A}$ and is generated by some random process. Based on the given sequence, the algorithm approximates the ideal predictor of the random process in the form of a hidden Markov model: The CSSR algorithm outputs the state space $\mathcal{S}$ of a discrete-time Markov chain (DTMC; see [18] for theoretical background) and its transition probabilities, where at each state transition, i.e., time step, the DTMC outputs a symbol from the alphabet $\mathcal{A}$. In the context of computational mechanics, this Markov chain is called ϵ-machine. For every time step, let S denote the random variable of the current state of the ϵ-machine, S' the random variable of the successor state of S, and A the random variable of the symbol which is output during the transition from state S to S'. For any two states $s, s' \in \mathcal{S}$ and any symbol $a \in \mathcal{A}$, $P[S' = s', A = a | S = s]$ denotes the probability that the DTMC leaves state s and goes to state s' while emitting the symbol a.

The CSSR algorithm additionally requires the parameter $L_{\max} \in \mathbb{N}_0$: To approximate the ideal predictor of the time series, the CSSR algorithm investigates subsequences of length L within the input sequence, starting from $L = 0$. L is subsequently increased by one until $L = L_{\max}$. Thus, $L_{\max}$ corresponds to the maximal amount of time steps of the past that are incorporated by the CSSR algorithm to approximate the probability distribution of the next output symbol. The choice of $L_{\max}$ is based on a trade-off between the required computation time and the accuracy of the resulting ϵ-machine.

5 A Generic Method to Derive Local Interaction Strategies

In the following, we propose our generic method that consists of four steps. The approach is briefly discussed at the end of Sec. 5. The method is carried out as follows:

Step 1*: Modeling the System* Our method assumes that the system is modeled as a group of entities that interact by exchanging information at discrete time steps. Each entity of the system receives input from a countable set $\mathcal{I}$ and generates some output from a countable set $\mathcal{O}$. The entities' strategies map any input $i \in \mathcal{I}$ to some output $o \in \mathcal{O}$. The time-series analysis in Step 3 requires that the model can be evaluated by simulation. Besides these assumptions, our method does not impose any further restrictions on the modeling process. There are standard modeling formalisms that we expect to be particularly suitable to our approach since the local interaction of the entities play a central role, e.g., in game-theoretic models, cellular automatons, and graph-based models.

Step 2*: Specifying the Laplace's Demon* One entity of the system is chosen to be the LD and is modified as follows: The LD becomes an omniscient entity, i.e., it is equipped with global knowledge. Based on global knowledge, the LD's strategy can be optimized to foster the preferred macro-level behavior.

Step 3*: System Simulation and Analysis of the Laplace's Demon's Behavior* Based on the model derived in Steps 1 and 2, a simulation environment is set up to simulate the system including the LD. The simulation is executed and the input/output (I/O) behavior of the LD is analyzed: For each time step $t \in \mathbb{N}$, we denote with $(i_t, o_t) \in \mathcal{I} \times \mathcal{O}$ the I/O behavior of the LD, where i_t is the input which leads to the output o_t of the LD. A simulation of $N \in \mathbb{N}$ time steps generates a sequence $(i_t, o_t)_{t=1,\ldots,N}$, which can be analyzed by the CSSR algorithm as a time series with alphabet $\mathcal{I} \times \mathcal{O}$. Then, the output of the CSSR algorithm is an ϵ-machine with state space $\mathcal{S}$ and transition probabilities $P[S' = s', (I, O) = (i, o)|S = s]$ for all $s, s' \in \mathcal{S}$ and $(i, o) \in \mathcal{I} \times \mathcal{O}$, where S is the random variable of the current state of the ϵ-machine, S' the random variable of the successor state of S, and (I, O) the random variable of the I/O behavior the LD exposes at the state transition from S to S'. Thus, in other words, the derived ϵ-machine describes the I/O behavior of the LD.

Step 4*: Imitating the Laplace's Demon* This step derives a vicinity-dependent strategy $\mathcal{L}$ that imitates the omniscient LD. To derive $\mathcal{L}$, the ϵ-machine obtained in Step 3 is used.

$\mathcal{L}$ adopts the state space $\mathcal{S}$ of the ϵ-machine and imitates the LD's behavior as follows: At first, $\mathcal{L}$ randomly chooses its initial state (e.g., according to the stationary probability distribution of the ϵ-machine). If $\mathcal{L}$ is in state $s \in \mathcal{S}$ and receives some input $i \in \mathcal{I}$, then $P[S' = s', O = o|S = s, I = i]$ is the probability that $\mathcal{L}$ changes its state to $s' \in \mathcal{S}$ and outputs $o \in \mathcal{O}$. This probability can be calculated by conditioning on the event $\{I = i\}$ and using the definition of the conditional probability:

$$P[S' = s', O = o|S = s, I = i] = \frac{P[S' = s', (I, O) = (i, o)|S = s]}{P[I = i|S = s]} \,.$$

The probability $P[I = i|S = s]$ in the denominator can be calculated by marginalization:

$$P[S' = s', O = o|S = s, I = i] = \frac{P[S' = s', (I, O) = (i, o)|S = s]}{\sum\limits_{s' \in \mathcal{S}} \sum\limits_{o \in \mathcal{O}} P[S' = s', (I, O) = (i, o)|S = s]} \,, \quad (2)$$

where the probabilities on the right side of (2) are the known transition probabilities of the ϵ-machine obtained in Step 3.

Since the ϵ-machine describes the I/O behavior of the omniscient LD, $\mathcal{L}$ effectively reproduces the LD's behavior while being purely vicinity-dependent, i.e., its output behavior is determined only by its current state s, its input i, and its transition probabilities.

Remarks The resulting vicinity-dependent strategy is tailored towards the chosen model and its parameters, i.e., the strategy might lead to a different behavior when applied in a different environment as specified in Step 1 of the method. Hence, it is crucial to take care that the chosen model and its parameters accurately reflect the behavior of the system under investigation.

Nevertheless, resulting strategies can easily be adapted to a different environment or preferred macro-level behavior by applying the method again with refined omniscient strategies, a modified model, and/or parameter set.

Since the derived strategy is available in form of a DTMC, Markovian analysis can be used to evaluate its behavior.

We are completely aware of the fact that the simulation of systems with numerous interacting entities cannot guarantee that all possible states are investigated since such systems often expose a very complex and sometimes even chaotic behavior. Nevertheless, extensive simulation in Step 3 of the method results in a vicinity-dependent strategy which behaves optimal at least in the simulated cases.

Our method can also be applied iteratively. The derived strategy can be fed back to the environment leading to a refined strategy during the next iteration.

6 Application of the Method to the Iterated Prisoner's Dilemma

The method proposed in Sec. 5 is now applied to find two vicinity-dependent micro-level strategies for the IPD (cf. Sec. 3), which have different macro-level goals: The strategy $\mathcal{L}_\mathrm{A}$ imitates the *altruistic* LD, which fosters global cooperation. In contrast, the strategy $\mathcal{L}_\mathrm{E}$ imitates the *egoistic* LD, which has the aim of maximizing its own payoff. The method is applied as follows:

Step 1*: Modeling the System* In our model, we investigate the behavior of a single player, which plays the IPD with another player we call the opponent. The opponent randomly chooses one strategy from AC, AD, Tit-for-Tat, p-Tit-for-Tat, and Random: The strategy AC always cooperates and is, therefore, a naïve friendly strategy which gets easily exploited by a defector. AD always defects and is an archetype of an egoistic and distrustful strategy. Tit-for-Tat initially cooperates and then always makes the same choice as the other player during the last iteration. Tit-for-Tat is a smart strategy: It is friendly since it initially wants to cooperate. But in contrast to AC, it is not exploited if its opponent only defects. If its opponent tries to cooperate again, then it immediately forgives. With p-Tit-for-Tat we denote the pessimistic equivalent to Tit-for-Tat, which initially defects. Random is a strategy that at each iteration chooses C or D with a probability of 0.5. From the viewpoint of Random's opponent, Random is an unreliable strategy in the sense that no action leads to mutual cooperation or defection or any other preferred behavior.

In our model, the opponent randomly chooses one of the five mentioned strategies according to a uniform distribution. Then, the IPD starts: At each time step, the player plays the prisoner's dilemma with its opponent. At the end of each iteration, the player is informed of the choice of the opponent, and vice versa. Hence, the input and the output alphabet for both players is $\mathcal{I} = \mathcal{O} = \{\mathrm{C}, \mathrm{D}\}$. At each time step and with a probability $p_c \ll 1$ ($p_c \in (0, 1)$), the IPD is stopped and the opponent randomly chooses a different strategy.

Then, the IPD is restarted. Thus, for each chosen strategy, the IPD is carried out in sequence for $1/p_c$ time steps in the mean.

Step 2*: Specifying the Laplace's Demon* We investigate the behavior of two LDs: The altruistic and the egoistic LD. Both are made omniscient by informing them of the strategy of their opponent. Depending on this knowledge, the LDs implement a micro-level strategy which fosters the preferred macro-level behavior.

The altruistic LD follows a smart strategy which fosters cooperation: If the opponent is willing to cooperate, i.e., with AC, TIT-FOR-TAT, or P-TIT-FOR-TAT, then the LD implements the strategy AC. In contrast, if the opponent always defects, then the LD implements AD, which hinders the opponent to exploit the LD and implicitly fosters global cooperation by making defection less profitable than mutual cooperation. In confrontation with RANDOM, the LD implements TIT-FOR-TAT, which in the mean keeps the balance between cooperation and defection.

The egoistic LD has the aim of maximizing its own payoff. To do so, it answers with AD if its opponent implements AC, AD, or RANDOM. In contrast, if its opponent implements a smart strategy, i.e., TIT-FOR-TAT or P-TIT-FOR-TAT, it answers with AC.

Step 3*: System Simulation and Analysis of the Laplace's Demon's Behavior* The IPD iterations, as described in Step 1, are carried out 2^{20} times while p_c is set to 0.01. Thus, in the mean, the investigated LD plays the IPD for 100 times before the IPD gets restarted with another strategy of the opponent.

Each iteration step generates a pair $(i_t, o_t) \in \{\mathrm{C}, \mathrm{D}\}^2$, where i_t is the input that leads to the answer o_t of the LD at time step t. The generated sequence of values from the alphabet $\mathcal{A} = \{\mathrm{C}, \mathrm{D}\}^2$ is then analyzed by the CSSR algorithm. The algorithm's parameter $L_{\max}$ is chosen to be 1 since for all investigated strategies of the opponent, at most one step of the past is relevant for making their choice.

For both LDs, the altruistic and the egoistic, the CSSR algorithm outputs an ϵ-machine with three states and twelve transition probabilities. For all of these twelve transition probabilities, a 99%-confidence interval is calculated based on 100 applications of the method. The radius of the largest confidence interval evaluates to 0.001 which shows that 2^{20} iterations can be assumed to be enough to obtain reasonably reliable results.

Step 4*: Imitating the Laplace's Demon* From the ϵ-machines obtained in Step 3, the vicinity-dependent IPD strategies $\mathcal{L}_\mathrm{A}$ and $\mathcal{L}_\mathrm{E}$ can be derived by adopting the ϵ-machines' state space and calculating the strategies' transition probabilities according to (2).

The strategy $\mathcal{L}_\mathrm{A}$ is shown on the left and $\mathcal{L}_\mathrm{E}$ on the right side of Fig. 1: The states of each strategy are represented by nodes, numbered from 0 to 2. An arc from state s to s' ($s, s' \in \{0, 1, 2\}$) is labeled with $i|p|o$: If the opponent's last choice was $i \in \{\mathrm{C}, \mathrm{D}\}$, then the strategy changes its state from s to s' with transition probability $p \in (0,1]$ and answers with $o \in \{\mathrm{C}, \mathrm{D}\}$ in the current iteration of the IPD.

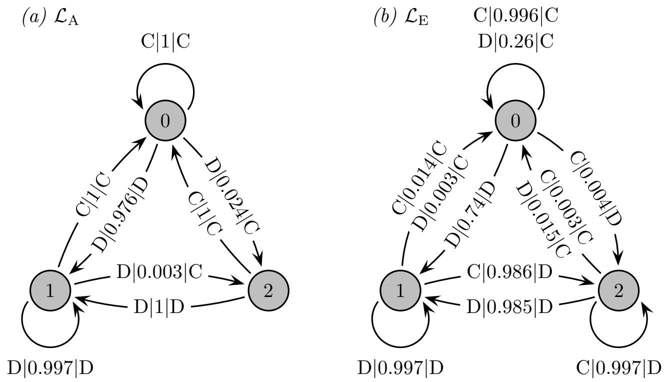

Fig. 1. The systematically derived vicinity-dependent strategies for the IPD: *(a)* shows the altruistic strategy $\mathcal{L}_\mathrm{A}$ and *(b)* the egoistic strategy $\mathcal{L}_\mathrm{E}$. The nodes correspond to the states of the strategies. An arc from node s to s' ($s, s' \in \{0,1,2\}$) is labeled with $i|p|o$, where $p \in (0,1]$ is the probability that the strategy chooses $o \in \{\mathrm{C}, \mathrm{D}\}$ and goes to state s' if its current state is s and the opponent chose $i \in \{\mathrm{C}, \mathrm{D}\}$.

6.1 Discussion of the Results

In Fig. 1, it can be seen that $\mathcal{L}_\mathrm{A}$ exposes a similar behavior to Tit-for-Tat: If the opponent cooperates, it answers with cooperation and goes to State 0. If the opponent defects, it also defects with a high probability and goes to State 1. Moreover, $\mathcal{L}_\mathrm{A}$ implements a forgiveness mechanism, just like the "tit for tat with forgiveness" strategy proposed by Anatol Rapoport (cf. [19]): With a relatively low probability, $\mathcal{L}_\mathrm{A}$ cooperates even if the opponent defects and goes to State 2. In State 2, $\mathcal{L}_\mathrm{A}$ waits for the reaction of the opponent: Cooperation leads to State 0 and defection to State 1. By this mechanism, $\mathcal{L}_\mathrm{A}$ is able to find out whether a defecting opponent is willing to cooperate in principle. This is especially helpful for establishing cooperation if the opponent implements the initially defecting p-Tit-for-Tat strategy.

The egoistic strategy $\mathcal{L}_\mathrm{E}$ in Fig. 1 can be described similarly: In State 0, $\mathcal{L}_\mathrm{E}$ cooperates with a high probability if its opponent cooperates. However, from time to time $\mathcal{L}_\mathrm{E}$ answers with defection to find out if the opponent implements AC and can, thus, be exploited. This exploitation is done in State 2, where cooperation is answered with defection with a high probability. If the opponent answers with defection, $\mathcal{L}_\mathrm{E}$ goes to State 1 from any state and defects with a high probability. Hence, State 1 corresponds to mutual defection. However, if $\mathcal{L}_\mathrm{E}$'s opponent implements Tit-for-Tat or p-Tit-for-Tat, then mutual cooperation leads to a higher personal payoff than mutual defection. Hence, $\mathcal{L}_\mathrm{E}$

implements a forgiveness mechanism and cooperates on defection with a low probability.

Most interestingly, both derived strategies are able to reconstruct the same knowledge that was granted to the omniscient LDs explicitly while being purely vicinity-dependent. $\mathcal{L}_{\mathrm{A}}$ is able to find out if the opponent is willing to cooperate. $\mathcal{L}_{\mathrm{E}}$ finds out if the opponent can either be exploited, implements a smart strategy (i.e., Tit-for-Tat or p-Tit-for-Tat), or always defects. Based on this information, the derived strategies adjust their behavior to either establish cooperation or to maximize the personal payoff.

It is worthwhile to note that the transition probabilities of both strategies reflect the chosen model assumptions and parameters. This can, for example, be seen in $\mathcal{L}_{\mathrm{E}}$'s transition between states 2 and 0 labeled with C|0.003|C, which does not lead to maximal payoff if the opponent implements AC. In this case, cooperation should be answered with defection with a probability of 1. This is because $\mathcal{L}_{\mathrm{E}}$ assumes that the opponent's strategy changes with a probability of $p_c = 0.01$ according to a uniform probability distribution on the set of the other strategies. Consequently, changing this probability distribution or the value of p_c leads to other transition probabilities of the strategy $\mathcal{L}_{\mathrm{E}}$ or, respectively, $\mathcal{L}_{\mathrm{A}}$.

6.2 Steady-State Analysis

Since the derived vicinity-dependent strategies are obtained in form of a DTMC, results obtained by steady-state Markovian analysis provide measures to compare the strategies $\mathcal{L}_{\mathrm{A}}$, Tit-for-Tat, and $\mathcal{L}_{\mathrm{E}}$. Each of the three strategies play the IPD against the strategies AC, AD, Tit-for-Tat, p-Tit-for-Tat, and Random. For all encounters, steady-state analysis is applied to obtain the expected payoff per iteration of the IPD. To obtain concrete values, the payoff matrix **U** given in (1) is considered. The numerical results are summarized in Tab. 1. The table is subdivided into three columns: The first column lists the five opponents, the second column consists of the expected payoffs for the strategies $\mathcal{L}_{\mathrm{A}}$, Tit-for-Tat, and $\mathcal{L}_{\mathrm{E}}$, and the third column lists the expected payoffs for the respective opponent.

Table 1. Results from the steady-state analysis of $\mathcal{L}_{\mathrm{A}}$, Tit-for-Tat, and $\mathcal{L}_{\mathrm{E}}$

	Payoff			Payoff (Opp.)		
	$\mathcal{L}_{\mathrm{A}}$	TfT	$\mathcal{L}_{\mathrm{E}}$	$\mathcal{L}_{\mathrm{A}}$	TfT	$\mathcal{L}_{\mathrm{E}}$
AC	3.0	3.0	3.571	3.0	3.0	1.286
AD	0.997	1.0	0.996	1.009	1.0	1.012
Tit-for-Tat	3.0	3.0	1.336	3.0	3.0	1.336
p-Tit-for-Tat	3.0	2.0	1.336	3.0	2.0	1.336
Random	1.993	2.0	2.477	2.020	2.0	0.569

It can be seen that $\mathcal{L}_{\mathrm{A}}$ behaves similar to Tit-for-Tat and clearly outperforms Tit-for-Tat if confronted with p-Tit-for-Tat since $\mathcal{L}_{\mathrm{A}}$ is able to establish cooperation. If Tit-for-Tat plays against p-Tit-for-Tat, mutual cooperation cannot be established, since the two strategies alternately cooperate and defect. When playing against AD, $\mathcal{L}_{\mathrm{A}}$ receives a lower expected payoff than Tit-for-Tat. This is due to the fact that $\mathcal{L}_{\mathrm{A}}$ once in a while tries to establish cooperation. The numerical results for $\mathcal{L}_{\mathrm{A}}$ show that it indeed fulfills the requirements of a strategy that fosters global cooperation: If its opponent is willing to cooperate, $\mathcal{L}_{\mathrm{A}}$ is always able to establish cooperation. If its opponent always defects, then $\mathcal{L}_{\mathrm{A}}$ answers with defection, which implicitly fosters global cooperation by making AD a less profitable strategy than a cooperating strategy like Tit-for-Tat.

$\mathcal{L}_{\mathrm{E}}$ is able to exploit AC and Random. If confronted with AD, $\mathcal{L}_{\mathrm{E}}$'s expected payoff is slightly lower than that of Tit-for-Tat because it cooperates with a low probability to check if the opponent's strategy has changed. Tit-for-Tat and p-Tit-for-Tat cannot be exploited by $\mathcal{L}_{\mathrm{E}}$. However, $\mathcal{L}_{\mathrm{E}}$ still tries to exploit the opponent from time to time. This again leads to a lower expected payoff compared to Tit-for-Tat.

7 Conclusion and Future Work

In this paper, we have shown the advantages and disadvantages of implementing purely vicinity-dependent or omniscient entities within the micro-level of self-organizing systems. Based on this investigation, we proposed a generic top-down approach to systematically derive vicinity-dependent strategies based on the specification of the preferred macro-level behavior.

Our approach simplifies the development of self-organizing systems in the sense that suitable interaction strategies can first be defined based on global knowledge, i.e., information of the self-organizing system's global state and/or the other entities' strategies. From these omniscient strategies, our method then systematically generates vicinity-dependent strategies. The resulting strategies can then be evaluated using standard Markov chain-based analysis techniques and can also be implemented easily into the entities of the self-organizing system.

The proposed method was proven to be applicable in a game-theoretic setting, where an altruistic and egoistic strategy for players of the iterated prisoner's dilemma could be obtained.

However, our method is presented independently of the applied modeling formalism. We expect the method to be applicable also to other modeling formalisms that focus on the local interaction of the micro-level entities, e.g., cellular automatons or graph-based models, and more complex scenarios. This has to be investigated in future work along with scalability issues that arise when engineering large-scale SOSs. We also foresee an extension to our method which allows to specify and evaluate scenarios comprising more than one LD with possibly different goals. The outcome of these investigations will also lead to further application examples and to more concrete advises on how to build suitable models to be used with the proposed method.

References

1. Caro, G.D., Dorigo, M.: AntNet: a mobile agents approach to adaptive routing. Technical Report IRIDIA/97-12, Université Libre de Bruxelles, Belgium
2. Cohen, B.: Incentives build robustness in BitTorrent (2003)
3. Jiang, J., Bai, H., Wang, W.: Trust and cooperation in peer-to-peer systems. In: GCC (1). (2003) 371–378
4. Lai, K., Feldman, M., Stoica, Chuang, J.: Incentives for cooperation in peer-to-peer networks. In: Workshop on Economics of Peer-to-Peer Systems. (2003)
5. Jun, S., Ahamad, M.: Incentives in BitTorrent induce free riding. In: P2PECON '05: Proceeding of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems, New York, NY, USA, ACM Press (2005) 116–121
6. Axelrod, R.: The Evolution of Cooperation. Basic Books, Inc., Publishers, New York (1984)
7. Yew Chong, S., Humble, J., Kendall, G., Li, M., Yao, X.: The Iterated Prisoner's Dilemma: 20 Years On. Chapter 1. In: The Iterated Prisoner's Dilemma: 20 Years On. World Scientific (2006)
8. Shalizi, C.R.: Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata. PhD thesis, University of Wisconsin (2001) Supervisor: Martin Olsson.
9. Shalizi, C.R., Shalizi, K.L.: Blind construction of optimal nonlinear recursive predictors for discrete sequences. In: AUAI '04: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, Arlington, Virginia, United States, AUAI Press (2004) 504–511
10. Perkins, C., Bhagwat, P.: Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications. (1994) 234–244
11. Clausen, T., Jacquet, P.: Optimized link state routing protocol (OLSR) (2003)
12. Diamantopoulos, F., Economides, A.A.: A performance study of DSDV-based CLUSTERPOW and DSDV routing algorithms for sensor network applications. In: ISWPC 2006: Proceedings of the 1st Internation Symposium on Wireless Pervasive Computing, IEEE Press (2006) 1–6
13. Eugster, P.T., Guerraoui, R., Kermarrec, A.M., Massoulieacute;, L.: Epidemic information dissemination in distributed systems. Computer **37**(5) (2004) 60–67
14. Jelasity, M., Babaoglu, O.: T-Man: Gossip-based overlay topology management. In: Proceedings of Engineering Self-Organising Applications (ESOA'05). (July 2005)
15. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and replication in unstructured peer-to-peer networks. In: ICS '02: Proceedings of the 16th International Conference on Supercomputing, New York, NY, USA, ACM (2002) 84–95
16. Haas, Z.J., Halpern, J.Y., Li, L.: Gossip-based ad hoc routing. IEEE/ACM Trans. Netw. **14**(3) (2006) 479–491
17. Osborne, M.J., Rubinstein, A.: A Course in Game Theory. The MIT Press (July 1994)
18. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S.: Queueing Networks and Markov Chains. 2nd edn. John Wiley & Sons, Inc., Hoboken, New Jersey (2006)
19. Rapoport, A., Cammah, A.M.: Prisoner's Dilemma: A Study in Conflict and Cooperation. University of Michigan Press (1965)