# Comparison of Learning to Cooperate and of The Evolution of Cooperation*

Jörg Wellner
Technical University of Chemnitz, Computer Science
D-09107 Chemnitz, Germany
jwe@informatik.tu-chemnitz.de

Sigmar Papendick
University of Konstanz, Department of Sociology
Universitätsstr. 10, PF 5560
D-78464 Konstanz, Germany
Sigmar.Papendick@uni-konstanz.de

Werner Dilger
Technical University of Chemnitz, Computer Science
D-09107 Chemnitz, Germany
wdi@informatik.tu-chemnitz.de

## ABSTRACT

Adaptation of multi-agent systems is either achieved by individual learning of single agents, or by an evolution of the entire agent system. Both approaches may have advantages in several respects. However, there are hardly any attempts to compare both approaches on the same problem in order to draw reasonably comparable conclusions. In this paper, we have chosen a sociologically inspired agent model to investigate the performance of a reinforcement learning agent architecture, and of a frame-like agent architecture which is subjected to evolution. Adaptation concerns acquiring a shared symbol system in order to coordinate actions in a simple cooperation game.

## 1. INTRODUCTION

In this paper we compare two different ways of achieving a population of coordinated agents. One approach is based on a simple individual reinforcement learning mechanism, the other is based on the evolution of frame-like structures. Both scenarios are set up under similar conditions in order to achieve a sufficient frame for comparison. As a general scenario we have chosen a coordination problem which may be solved by agents only when they become able to interpret exchanged messages in a correct way. The more agents share the same interpretation, the greater is the chance of cooperation.

There exists a reasonable amount of work in the field of the evolution of communication and cooperation (see e. g. [7], [2], or [9]), mainly inspired by sociobiology and ethology. In the same context, only a fiew approaches of learning a communication system (see e. g. [10] or [11]) exist. This is less surprising if one remembers that most communication skills of animals are innate (see [4]). Since innate communication skills are less flexible than those adapted during individual life time the question arises what the advantages and trade-offs of each approach are, especially in relation to multi-agent systems. However, since work on multi-agent systems follows human communicative and interactive behavior rather than studies on animal behavior, we have chosen a sociological background for our simulation scenario. Moreover, we are not only interested in learning or evolving a shared symbol system by agents. Since the coordination of agents is still a prominent problem in multi-agent research, we focus in this paper on achieving coordinated action sequences, based on an emerging symbol system.

The framework of our studies is given by Luhmann's sociological systems theory (see [6], or for a short introduction see [5]). This theory adopts autopoietic theory ([8]) to society and treats human society as a closed system, mainly based on communication. Cooperation is explained with respect to the so-called problem of double contingency. This problem can be described in a simplified way by the black box metaphor (two agents are black boxes to each other and are not able to look "inside the other's head"). Taking this problem seriously also in multi-agent research, may lead to new approaches regarding problems related to achieving coordinated actions, without pre-defining an interpretation (meaning) of exchanged messages.

In the next section, we introduce a simple model for investigating the performances of the two different agent approaches. The learning type agents, which are based on individual reinforcement learning is presented in Section 3. Section 4 is devoted to the evolutionary approach, which is based on evolving frame-like structures. A comparison of both approaches is done in Section 5, and Section 6 concludes the paper.

## 2. THE GENERAL MODEL

A simulation consists of a large number of trials of a cooperation game which we called the "Planter-and-Harvester-Game" for simplicity. We introduce two different types of agents, with respect to their ability to change the environment. There are also two types of actions that change the state of the environment in an effective way, namely "planting" and "harvesting" complementing each other. Plant

*This work is supported by the *Deutsche Forschungsgemeinschaft* under grant number DI 452/10-1.

agents, called $Planter$ (or agent type 0) can perform only plant actions effectively, harvest agents, called $Harvester$ (or agent type 1) can perform only harvest actions effectively. At the beginning of a game the environment $U$ is always in state $U_s = 0$. A plant action $Plant_I$ – performed by a $Planter$ – transforms the environment into state $U_t = 1$, a harvest action $Harvest_I$ – performed by a $Harvester$ – transforms it into the final state $U_e = 2$. In more complicated games the final state may be $U_e > 2$ assuming action sequences $Plant_I$, $Harvest_I$, $Plant_{II}$, and so on. Action $Plant_I$ in state $U = 1$ has no effect with regard to the state of the environment, similarly action $Harvest_I$ in state $U = 0$. In general, the transformation of the environment state $U_t$ at time $t$ that a plant agent may perform by action $a_t$ is defined by

$$[(U_t = i) \wedge ((i \bmod 2 = 0) \wedge (type = 0)) \wedge (a_t = i)] \Longrightarrow$$

$$U_{t+1} := i + 1$$

and for harvest agents by

$$[(U_t = i) \wedge ((i \bmod 2 = 1) \wedge (type = 1)) \wedge (a_t = i)] \Longrightarrow$$

$$U_{t+1} := i + 1 \; .$$

In any other situation the environment remains in state $i$ ($U_{t+1} = i$).

At the beginning of a game two agents are randomly selected from the population, and one of them is chosen as the start agent. This agent begins by sending a message $M_0$. The other agent receives this message and does both, it – the first time – performs an action $a_1$ and sends another message $M_1$ to the start agent. Then, the first agent performs an action $a_2$ and sends a message $M_2$ to the second agent, and so on. A round is defined by a successive sequence of performing one action and generating a message for each of the two agents. Both types of agents have the same repertoire of actions regardless of the efficiency: apart from plant and harvest actions they have a $Null$-action without any effect, a sanctioning action $Bite$, an action $Exit$, and an action $Replace$. The later action affects the opponent agent in the way that it gets replaced by a other agent, randomly selected from the population. This may increase the general possibility for a successful coordination. A game may end with three different outcomes: an agent performed the $Exit$ action, the environment reached the final state $U_e$, or the number of rounds in the games exceeded the defined threshold $rounds$.

There is a predefined set of symbols $\mathcal{S} = \{0, 1, 2, ..., S_{max}\}$. A message consists exactly of one of these symbols. A symbol itself has no meaning to an agent, there is no predefined semantic at all.

A game ends successfully if the environment was transformed into the final state. In this case, the last two agents participating in the game get a certain amount $E^\star$ of "energy". In other cases there is no energy payoff. Every action that an agent performs consumes a specified amount of energy of the agent. There are low cost actions ($Null$, $Exit$, and $Replace$) and high cost actions ($Plant_x$, $Harvest_x$). For a low cost action the agent consumes energy $E_l > 0$, for a high cost action $E_l + E_h$, $E_h > 0$. The cost of the action $Bite$

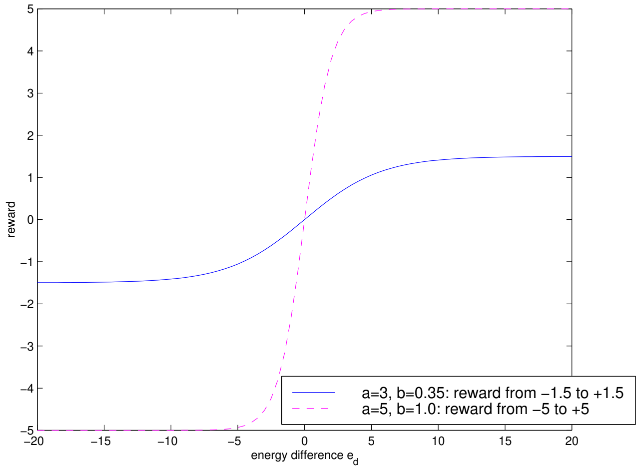


**Figure 1: Two examples of the reward generating function.**

is $E_l + E_b$, $E_b > 0$. This action affects the other agent in the way that the "bitten" agent looses pain energy $E_p > 0$. At the beginning of an agent's life time its energy is set to $E = E_s > 0$, its start energy. If $E$ ever falls below 0, the agent dies, i. e., the agent is removed from the population.

An agent does not know its own type nor perceives the type of another agent. They are black-boxes to each other. An agent perceives the message of another agent and – perhaps – some sensory input like the state of the environment or the fact of being bitten. In any case not all relevant aspects of the environment are known in the same way to all the participants. Agents must test different actions at different times and the only hint to whether an action or message was appropriate or not is given by a reward signal. This signal is always generated by the agent itself, based on the energy difference between two consecutive actions. A sigmoid function $f_r$ (see Equation (1)) generates the reward signal $r$ based on the energy difference $e_d$; a positive energy difference results in a positive reward, a negative difference results in a negative reward. The reward generating function $f_r$ is parametrized by two values, $a$ and $b$:

$$f_r = 2 * a * \frac{1}{1 + exp(-b * e_d)} - a \; . \tag{1}$$

Figure 1 shows two examples of the function. Thus, the individual learning of an agent is of reinforcement-learning type. This definition of a reward signal is a weak one, since it does not assume any intelligent observer (outside the agent) who generates a reward signal based on its knowledge with respect to correct actions.

Beside an energy value agents have an age $A$, which at the beginning of an agents life time is set to 0. Any time an agent is selected to play the game, its age will be incremented by 1. If the age reaches an individual maximum, $A_{max}$, the agent will be removed immediately from the population. At the start of the simulation, the population $P$ consists of a certain number of agents $P_s$. The number of agents during the simulation may shrink or grow, depending on the fitness of the agents. An agent may enter the population if there are at least two agents, whose age is above a value $A_{sex}$

and whose energy value is above a value $E_{sex}$. The two "parents" are selected by a "Roulette wheel" [3] from all possible parent agents based on their energy values. Once a successful breeding occurred, the two parent agents are prevented from reproduction for a certain period of time $t_{pause}$. The amount of gene material parents pass on to their child is described in the next section. The general schedule of the simulation is:

1. initialize start population $P_s$
2. do forever
   - select randomly two agents, $Agent_1$ and $Agent_2$, and $t := 0$, $U_0 := 0$
   - $Agent_1$ generates and sends a start message $M_0$
   - do
     - $t := t + 1$
     - $Agent_2$ receives previous message $M_{t-1}$ and generates action $a_t$ and message $M_t$
     - $t := t + 1$
     - $Agent_1$ receives previous message $M_{t-1}$ and generates action $a_t$ and message $M_t$

     until $U_t = U_s$ or $a_t = Exit$ or $t = 2 + rounds$
   - remove from or add agents to the population

Whenever the number of agents in the population $P_t$ falls below $P_s$, agents are randomly added to the population until $P_t = P_s$. If this situation happens it can be interpreted as restarting a simulation.

## 3. INDIVIDUAL Q-LEARNING

Due to our goal of finding out what type of learning mechanism enables a population of agents to develop a shared symbol system, we first consider an agent architecture which is solely based on individual learning and takes almost no advantage of evolutionary mechanisms. We have chosen a simple table-based, one-step $Q$-learning ([12]). Each agent has two $Q$-tables, one for the generation of the next action, and one for the generation of the next message. The selection of a new action is independent of the selection of the next message and vice versa.

The inner state of an agent, the $Q$-state, is based on the last three messages: the last received message $M_{t-1}$ and the previously received message $M_{t-3}$ and the last sent message $M_{t-2}$. For $S_{max}+1$ different symbols there are $(S_{max}+1)^3$ different states. Further we distinguish between different starting situations for the case $M_i = nil$, $i \in 0, 1, 2$. Thus, the state space of an agent consists of $[(S_{max}+1)^3 + (S_{max}+1)^2 + (S_{max}+1) + 1] \times U_e$ different states, taking the environmental state into account, too. The update of a $Q$-value $Q(a, s)$ of a state $s$ and an action $a$ is done by the $Q$-learning equation:

$$Q(a, s_t) \;:=\; Q(a, s_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q(a', s_{t+1}) - Q(a, s_t)]$$

where $\alpha$ and $\gamma$ are the usual learning parameters, $r_{t+1}$ the reward signal (given by Equation 1, and $\max_{a'} Q(a', s_{t+1})$ is the highest $Q$-value of state $s_{t+1}$. During the development of the population no evolution occurs, i. e., the $Q$-tables are initialized randomly and the parameters $\alpha$,$\gamma$,$\epsilon$[1], $a$, and $b$ are fixed[2]. No individually acquired knowledge is passed on to the offspring.

Figure 2 presents results of a simulation of 1000000 games. Agents were designed to sense the state of the environment, thus the message-based state space of 85, ($S_{max} = 3$), is multiplied by $U_e = 4$. The simulation starts off with 2

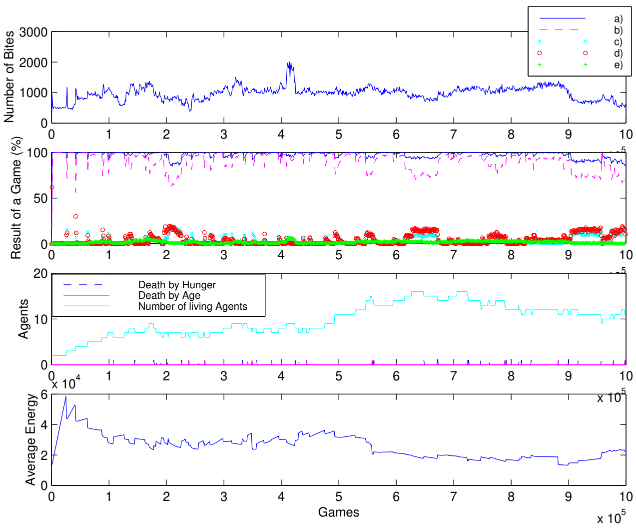


**Figure 2: $Q$-learning agents. Results averages 1000 games, apart from the *Bites*-graph, which shows the total number of bites in 1000 games. Result of the game (legend for the second graph is shown on top): $a$ : maximum possible success (counting the occurrence of a "correct" pairing of the agents); $b$ : the actually achieved success; $c$ : correctly performed *Exit*; $d$ : *Exit* in wrong situation; $e$ : stopped, because maximum rounds exceeded. $U_e = 4$, $S_{max} = 4$, $rounds = 40$, $E^\star = 5.0$, $E_l = 0.5$, $E_h = 0.25$, $E_b = 0.1$, $E_p = 0.2$, $E_s = 15000.0$, $A_{max} \in \{170000, \ldots, 200000\}$, $A_{sex} = 26000$, $t_{pause} = 20000$, $a = 1.0$, $b = 1.0$.**

agents, after about 25000 games a third agent enters the population. The results indicate that a small stable population of agents may improve its performance as long as no agent enters or leaves the population. However, a global functional coordination based on messages does not clearly emerge, as indicated by an analysis of main action-message sequences in Figure 3. The more agents are committed to the same sequences (see text of Figure 3 for the definition of a sequence) the greater is the overlap of interpretations of the used symbols. To be more precise, the sharing of symbol meanings increases if agents of the same type react and communicate in the same way in equal circumstances (type of other agent, state of the environment). Entering agents sometimes do not adapt to the "existing shared symbol system" and looses to much energy (death by hunger), because no stable population wide interpretation of sym-

[1] We have chosen an $\epsilon$-greedy action selection for the $Q$-learning algorithm.

[2] Our experiments have shown that evolution on basis of these parameters has no effect.

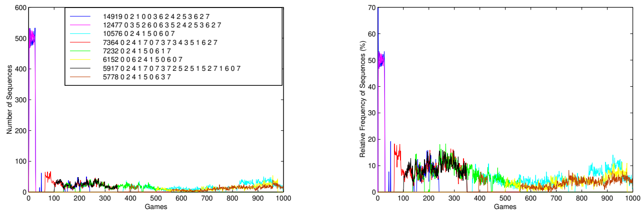

**Figure 3: $Q$-learning agents. The eight most frequent sequences, without a *Replace* action, averaged over 1000 games. There were 325964 successful sequences (sequences which includes a *Replace* action were not counted). Left: Absolute frequency (legend, in this order - in a row: total occurrence of sequence $M_0$, $M_1$ $a_1$ $M_2$ $a_2$ ..., e. g. first row: no. of seq. = 14919 $M_0$ = 0 $M_1$= 2 $a_1$ = 1 etc. ). Right: Relative Frequency. Coding of actions: 0 - *Null*, 1 - *Bite*, 2 - *Exit*, 3 - *Replace*, 4 - *Plant*$_1$, 5 - *Harvest*$_I$,.... (Because, we analyzed only sequences, which did not contain a *Replace*-action, and which were successfully, all these sequences end with action 7.)**

bols was achieved. Actions are performed rather based on sensory input than on received messages, but with lower performance than possible. Nevertheless, two interesting observations are worth to be noted. As shown in Figure 4, the two agents of the start period (up to 26000 games) did quite well coordinate their actions by two sequences, one sequence

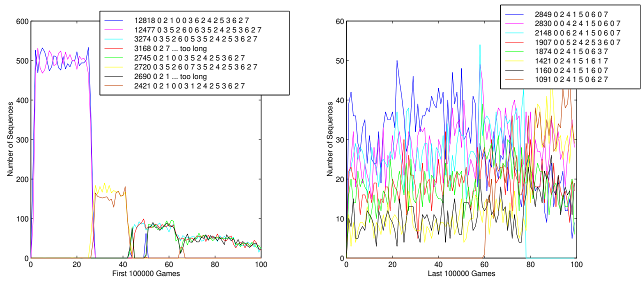

**Figure 4: $Q$-learning agents. The eight most frequent sequences of the first and of the last 100000 games.**

for the start of each agent. Apart from a different start of a sequence, the two sequences are equal. On the right side of Figure 4 the eight most frequent sequences are shown for the last 100000 games. Although there is no dominating sequence, all sequences are similar, and taking into account the type of the start agent (harvester agents start sequences 1,4,5,6,7, planter agents start 2,3,4), one realizes that some sequences differ only in one message. Since simulation conditions were fixed in the way that the start message $M_0$ was always set to 0, the two agent types learned to "answer" this start message in a type dependend way. Planters answer it by a 2, and harvesters by a 0.

There are many different aspects which may be chosen (apart from the previous sequence analysis) to investigate the communicative and environment related behavior. This analysis

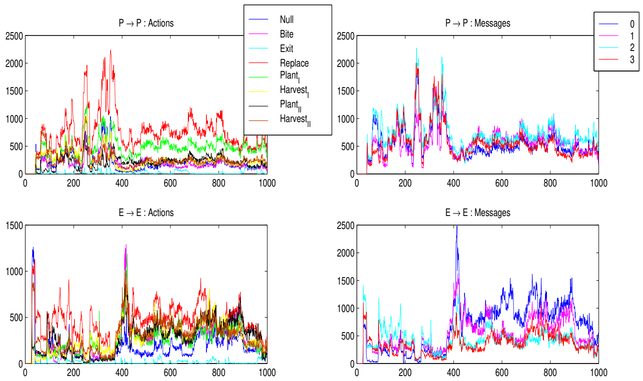


**Figure 5: $Q$-learning agents. Performed actions and sent messages of *Plant* agents and *Harvester* agents with respect to unfavorable agent grouping.**

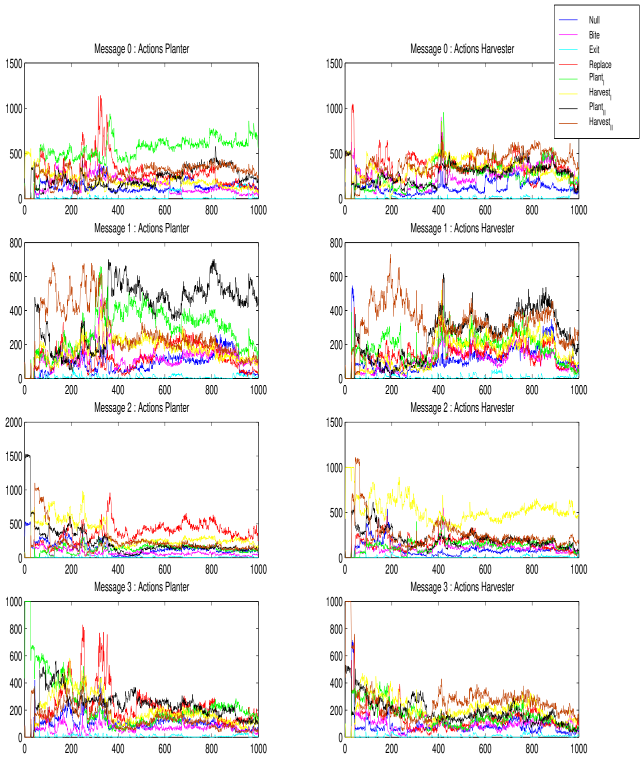


**Figure 6: $Q$-learning agents. The actions of *Plant* agents and *Harvest* agents which follow a certain message.**

can be further evaluated by regarding actions and messages in the case, agents are note favorably grouped (Figure 5), or by considering which actions were performed by agents as reaction of receiving a certain message (Figure 6). These analysis reveals again a non-optimal usage of the symbol system.

## 4. EVOLUTION OF FRAME-LIKE STRUCTURES

One aspect of the problem of double contingency is not explicitly modeled in the individual learning approach, namely the relevance of expectations in choosing an appropriate answer to a received message. Using previously sent messages in order to define an internal state on which an agent bases its reply and action does not capture the entire situation. Thus, we combine an internal state with the expectation of a received message. This results in a frame-like structure which will be executed on two levels. In a first step a set $F_i$ of frames is chosen based on the state of the environment. This step will be performed without any learning by the agent and is totally determined by the environment. In a second step the agent chooses one frame from the previously chosen set $F_i$. The selected frame will be executed resulting in an action $a_{t+1}$ and a new message $M_{t+1}$. A frame $F$ is defined with respect to a received message $M_r = M_t$ in the following way:

$$\begin{array}{llllll} \text{if} & M_r = M_{e1} & \text{then} & a := act_1 & \text{and} & M := mes_1 \\ \text{elseif} & M_r = M_{e2} & \text{then} & a := act_2 & \text{and} & M := mes_2 \\ \text{else} & & \multicolumn{4}{l}{\text{execute trouble frame} \quad F^T\ ,} \end{array}$$

where $a_{t+1} = a$ and $M_{t+1} = M$. The "trouble frame" $F^T$ will be executed in the case that the received message was neither $M_{e1}$ nor $M_{e2}$. This frame has a special structure, because it does not check the occurrence of a certain message, rather it checks whether the agent was bitten or not in order to determine the new action and message:

$$\begin{array}{lllll} \text{if} & bitten = true & \text{then } a := act_{T1} & \text{and} & M := mes_{T1} \\ \text{else} & & \text{then } a := act_{T2} & \text{and} & M := mes_{T2}\ . \end{array}$$

For every state of the environment the agent has $n_f \geq 1$ frames. The selection of a frame at time $t$ will be guided by a $Q$-value $Q_F$, i. e., reinforcement learning ([12]) takes place in order to choose an appropriate frame in a given (environmental) situation. The entire collection of frames for an agent by a given final state $U_e$ of the environment is:

| state (depends on $U$) | frame set $F_i = \{F_{(i,0)}, \ldots, F_{i,n_f-1}\}$ |
|---|---|
| 0: $U^\star$ | $F_{(0,0)}, \ldots, F_{(0,n_f-1)}$ |
| 1: $U = 0$ | $F_{(1,0)}, \ldots, F_{(1,n_f-1)}$ |
| 2: $U = 1$ | $F_{(2,0)}, \ldots, F_{(2,n_f-1)}$ |
| $\vdots$ | $\vdots$ |
| $U_e$: $U = U_e - 1$ | $F_{(U_e,0)}, \ldots, F_{(U_e,n_f-1)}$ |
| $U^T$ | $F_0^T, \ldots, F_{n_f-1}$ |

$U^\star$ is the frame set for an agent when it starts the communication by generating just a message $M_0$, and $U^T$ is the frame set for the trouble state.

Evolution is based on frames, agents do not change frames during their life time, they are just able to change the $Q$-value of a frame with respect to other frames inside the same frame set. At the beginning of the simulation, all frames of all agents are initialized randomly. In particular, variables $M_{e1}$, $M_{e2}$, $mes_1$, $mes_2$, $mes_{T1}$, and $mes_{T-2}$ are randomly chosen values from $\mathcal{S} = \{0, 1, 2, \ldots, S_{max}\}$, and variables $act_1$, $act_2$, $act_{T1}$, and $act_{T2}$ are randomly chosen values from $\mathcal{A} = \{Null, Bite, Exit, Replace, Plant_I, Harvest_I, Plant_{II}, \ldots\}$. Inheritance happens on the frame level, i. e., cross-over takes place *between* frames, not inside a frame (but inside a frame set). Individual parts of a frame are subjected to mutation. Therefore, e. g. part $M_{e1}$ or $act_2$ may get a new random value during a mutation process. $Q$-values are not passed on to offspring, and are set to small random values at the beginning of an agents life time.

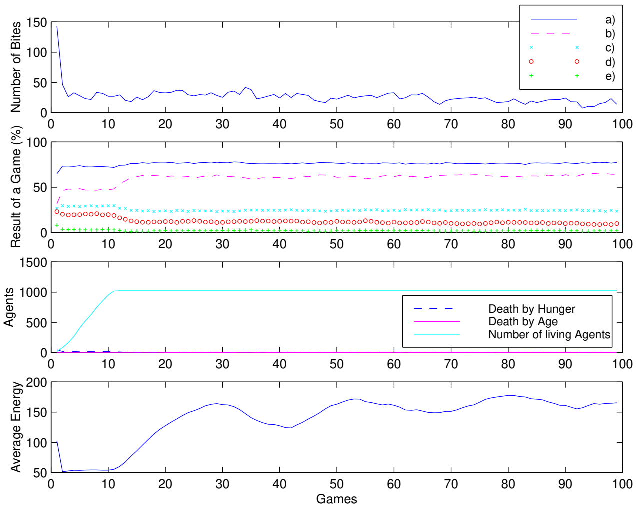


**Figure 7: Frame agents.** $U_e = 4$, $S_{max} = 3$, $rounds = 10$, $E^\star = 10.0$, $E_l = 0.5$, $E_h = 2.5$, $E_b = E_p = 0.1$, $E_s = 50.0$, $A_{max} \in \{550, \ldots, 800\}$, $A_{sex} = 20$, $t_{pause} = 20$, $a = 5.0$, $b = 1.0$, $n_f = 2$.

Figure 7 shows the general outcome of a simulation based on frame type agents. The maximum number of agents was set to 1024. The simulation started with 3 agents and as long as the number of agents was below 15 a higher energy pay off $E^\star$ was given for success than indicated in the figure (to support an onset of evolution). The number of agents grows rapidly until the limit is reached. Later, evolution still takes place optimizing the frame structures. This may result for example in changing cooperation sequences, or in a "competition" of different sequences (cf. Figure 8). Figure 8 shows the eight most frequent sequences of the entire simulation. In detail, they are:

| number (cf. Fig. 8) | number of occurrenc | sequence $M_0\ M_1\ a_1\ M_2\ a_2 \ldots$ |
|---|---|---|
| 1 | 160877 | 1 0 4 0 5 1 6 2 7 |
| 2 | 66551 | 2 0 4 0 4 0 5 1 6 2 7 |
| 3 | 37402 | 0 0 4 0 5 1 6 2 7 |
| 4 | 26721 | 0 1 5 0 4 0 5 1 6 2 7 |
| 5 | 19039 | 2 1 5 0 4 0 5 1 6 2 7 |
| 6 | 7118 | 0 0 4 0 4 0 5 1 6 2 7 |
| 7 | 6453 | 2 0 5 0 4 0 5 1 6 2 7 |
| 8 | 5734 | 2 1 7 0 4 0 5 1 6 2 7 |

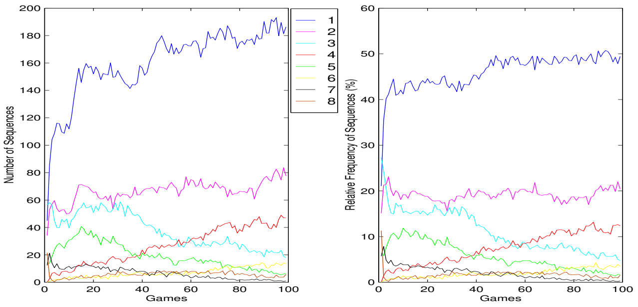


**Figure 8: Frame agents. The eight main sequences of the frame based evolution. Left: Absolute occurrence of sequences, right: relative occurrence of the sequences (in relation to 346727 successful sequences). The eight sequences occur 329895 times. See text for further explanations.**

The first sequence occurs 160877 times, out of 346727 successful sequences, without a *Replace*-action. In contrast to the previous simulation, frame agents were able to select the start message $M_0$.

The communicative behavior of agents becomes more and more regular. Because we have chosen $n_f = 2$ it is obvious that a frame set is assumed to contain exactly one appropriate frame for *Planters* and one for *Harvester*. An individual only has to explore which one is better suited. A detailed analysis of the communicative behavior reveals indeed that communication controls the behavior of agents. Thus, agents realize the type of the other agent and choose a useful action in the case that the other agent is of its own type (see Figure 9). Furthermore, we regard the question of which action is performed on the basis of a received message (see the analysis in Figure 10), and indeed, agents act in a foreseeable way, i. e., one agent may control by communication the action of the other agent. Further investigations have to be done, especially a more detailed analysis with respect to a combination of two or more dimensions of interest. For example, in which situation (state of the environment) a *Planter* reacts with which action after receiving message 0 (cf. left graph on top of Figure 10)?

## 5. COMPARISON

The parameters of each simulation were supposed to be similar, though optimal to some extend for each agent architecture. Because evolution "needs" frequent replacements of individuals, the life span of frame agents is much shorter than of the $Q$-learning agents. The energy payoff $E^\star$ for $Q$-learning agents is smaller, because $E^\star$ acts as a constraint to the agents in order to rely their action selection on received messages at all. Evolution of a working set of frame structures turned out to be not that easy. To be stable enough, an entire frame set has to reflexive, i. e., it is not enough to work for only one agent type, but simultaneously for both types. It is less likely, that totally different sets of frames match each other in the right way. However, our evolutionary approach is also based on individual learning, namely the selection of a frame from a given set at a certain

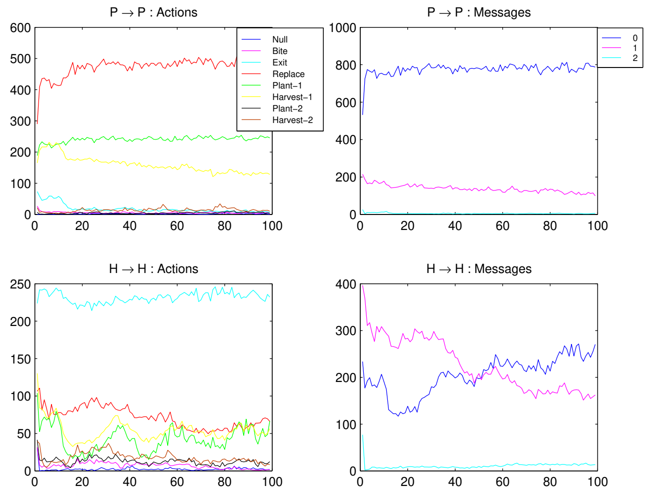


**Figure 9: Frame agents. Performed actions and sent messages of *Plant* agents and *Harvester* agents with respect to unfavorable agent grouping. *Planters* (top row) prefer the *Replaces* action, *Harvester* (bottom row) prefer the *Exit* action. Both types of agents realized that the other agent was not a "correct" partner and they did so without having any direct access to the type of an agent (neither to the own nor to the type of another agent).**

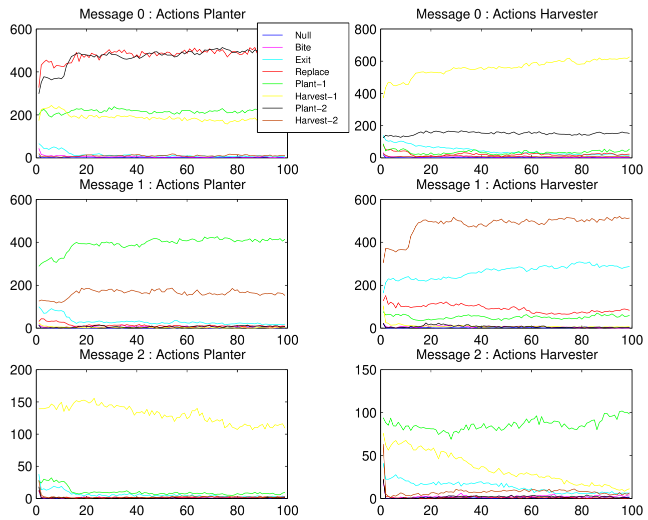


**Figure 10: Frame agents. The actions of *Plant* agents and *Harvest* agents which follow a certain message.**

time. This is a crucial aspect of the evolutionary scenario[3]. Other simulations with just one frame in every frame set, i. e., without individual learning, showed that an successful evolution of cooperative agents still takes place, but much less efficient. Sequences were longer and we did not found a permanent increase on the number of agents.

A striking difference of both simulations is, however, the permanent increase of the number of living frame-based agents. Only because the maximum number of agents were fixed to 1024, this increase reached a limit, otherwise the number would have grown further. As a result of this artificial limit, evolution worked more efficiently, because now, agents (i. e, different sets of frames) had to compete harder for a position in the set of living agents.

The following table summarizes our observations and stresses different features on the systems level:

| | **pure individual *Q* learning** | **evolution of frame structures** |
|---|---|---|
| scalability | poor | very good |
| individual flexibility | good | not so good |
| environment conditions (in relation to energy payoff) | less demanding | resource demanding (at least for the onset of evolution) |
| normative behavior | poor, rarely established | good |

Our results indicate, that pure individual reinforcement learning for agent systems may be suitable for cases in which only a few agents are needed and individual flexibility is favored. This coincides with previous results of other authors, who also applied reinforcement learning to multi-agent systems ([13], [14]). On the contrary, if one is interested in a huge population of agents, which may coordinate their behavior on a normative basis, we assume an evolutionary approach to be much more effective.

## 6. CONCLUSION

In this paper we have compared two different agent architectures, applied to the same coordination problem. The general scenario is about the acquisition of a shared symbol system in order to coordinate actions. Individual learning was achieved by simple reinforcement learning, evolution was based on frame-like structures. The frame-like architecture relies to some extent on individual reinforcement learning too. Our results indicate a greater individual flexibility of pure individual learning agents, a quicker shift toward a first adaptation, but a less promising scalability, and suboptimal efficiency. In contrast, the evolved structures are more resource consuming, especially in order to onset a promising evolution, but indicate a good scalability and a stringent tendency toward normative behavior of the agents. Hence, we argue to favor individual learning agents for problems that may be solved by a small number of agents, but an evolutionary approach to problems, where scalability and normative behavior are a key feature to the solution.

In subsequent work we will try to enhance efficiency of both agent architectures: $Q$-learning agents by second-order conditioning, based on exchange messages, and the frame-like architecture by extending individual learning capabilities.

[3] [1] claim evolutionary algorithms, applied to the evolution of cooperative agents, to be a core problem of evolutionary approaches.

## 7. REFERENCES


[1] Kenneth DeJong and William Spears. On The State of Evolutionary Computations. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 618–623, 1995.

[2] Ezequiel A. Di Paolo. An Investigation into the Evolution of Communication. *Adaptive Behavior*, 6(2):285–324, 1997.

[3] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Inc., Reading, Mass., 1989.

[4] Marc D. Hauser. *The Evolution of Communication*. A Bradford Book, The MIT Press, Cambridge, Mass., 1996.

[5] Niklas Luhmann. The Autopoiesis of Social Systems. In F. Geyer and J. van der Zouwen, editors, *Sociocybernetic Paradoxes*, page 172ff., London, 1986. Sage.

[6] Niklas Luhmann. *Social Systems*. Stanford University Press, Stanford, Ca., 1995.

[7] Bruce J. MacLennan and Gordon M. Burghardt. Synthetic Ethology and the Evolution of Cooperative Communication. *Adaptive Behavior*, 2(2):161–188, 1994.

[8] H. R. Maturana and F. J. Varela. *Autopiesis and cognition*. D. Reidel Publishing Company, Dordrecht, 1980.

[9] Jason Noble. Cooperation, Conflict and the Evolution of Communication. *Adaptive Behavior*, 7(3/4):349–369, 1999.

[10] Michael Oliphant. The Learning Barrier: Moving from Innate to Learned Systems of Communication. *Adaptive Behavior*, 7(3/4):371–383, 1999.

[11] Luc Steels and Frederic Kaplan. Collective Learning and Semiotic Dynamics. In Francesco Mondada Dario Floreano, Jean-Daniel Nicoud, editor, *Advances in Artificial Life: 5th European Conference, ECAL'99*, pages 679–688, Berlin, 1999. Springer-Verlag.

[12] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, Mass., 1998.

[13] Ming Tan. Multi-agent reinforcement learning: Independend vs. cooperative agents. In *Proceedings of the 10th International Conference on Machine Learning*, pages 330–337, San Francisco, 1993. Morgan Kaufmann.

[14] Gerhard Weiß. Learning to coordinate actions in multi-agent systems. In *Proceedings of the Thirteenth International Concerence on Artificial Intelligence*, pages 311–316, 1993.