Trading off Impact and Mutation of Knowledge by Cooperatively Learning Robots

Willi Richert, Bernd Kleinjohann, Lisa Kleinjohann

Intelligent Mobile Systems, University of Paderborn / C-LAB, Germany,
richert@c-lab.de

Abstract. We present a socially inspired approach that allows agents in Multi-Agent Systems to speed up their own learning process through communication. Thereby, they are able to trade off impact of knowledge by mutation dependent on the recent performance of the interacting agents. This is inspired by social interaction of humans, where the opinions of experts have greater impact on the overall opinion and are incorporated more exactly than those of newbies. The approach is successfully evaluated in a simulation in which mobile robots have to accomplish a task while taking care of timely recharging their resources.

1 Introduction

A lot of useful techniques exist for groups of agents that learn to behave optimally to reach a given task while adapting to their environment. Especially reinforcement learning (RL) [1, 2], where the agent does not need a predefined environment model and learns through reward and punishment that it receives from its environment, has been shown to be a viable solution for groups of behavior-based learning agents [3]. This approach has also been successfully used to learn in groups of agents to connect the individual agent's innate states to the proper behaviors it has to execute when being in the according state [4]. The success of such reinforcement learning systems depends in general on the careful design of the state and action space and the reward function. In many situations these have to be found out by careful analysis of the domain followed by a trial and error period — often leading to suboptimal solutions. In Multi-Agent Systems (MAS) the problem is on the one hand amplified since the interferences of the agents cannot be anticipated by the designer. This is especially true for environments, in which no central intelligence is available for coordination and optimization. On the other hand it can be relieved if proper learning methods are combined with robust mechanisms for spreading the learned knowledge between the agents.

In this case, however, the problem arises, how to integrate the received information into its own knowledge base if only sporadic communication possibilities exist. Our socially inspired approach that we describe in this paper contributes to this problem with the following properties:

Please use the following format when citing this chapter:

Richert, W., Kleinjohann, B., Kleinjohann, L., 2006, in IFIP International Federation for Information Processing, Volume 216, Biologically Inspired Cooperative Computing, eds. Pan, Y., Rammig, F., Schmeck, H., Solar, M., (Boston: Springer), pp. 189–198.

- The impact of knowledge learned by individuals is weighted in the communication process based on the recent performance of the participants, called "expert state". The more "expert" an agent is regarded the more influence it has on the final knowledge arbitration of the other participant.
- Mutation fosters new solutions based on the expert level of the participants. The less "expert" an agent is regarded the more mutated his information are transmitted.
- The number of expert agents is allowed to vary.
- In unknown or changing environments the performance of agents will decline resulting in more mutation until the first agent finds a way to perform better, that in turn increases his expert state and impact on the subsequent knowledge exchange. Thereby, this approach is robust to environmental change.

In our previous work we have shown how the knowledge transfer in societies of autonomous systems leads to the propagation of the most valuable information units that offer the biggest performance advantage [5, 6]. In that experiment we modeled the knowledge as a discrete sequence of actions which have influence on the agent's intrinsic performance evaluation. Based on the outcome of the imitated action sequences these were distributed in the agent society. Encouraged by these results, we use the imitation process in form of group learning to deal with continuous information units: the information, how the continuous state space is best to be discretized. The discretization in the current work is inspired by learning in human societies, where humans exchange their knowledge from time to time. Typically in this group learning process firstly the opinion of experts counts more than the opinion of less experienced group members. Furthermore, expert opinions tend to be more exactly integrated into the learning result. A good measure for experience are the age or lifetime of an agent and its accumulated performance. In this way, the propagation of useful knowledge in form of information units is not only dependent on the current state of the imitated agent, but also on its lifetime achievement. In this vein, we are approaching the optimization of the state space from the memetic point of view according to Dawkins [7, 8].

We evaluate the approach in a mobile robot application that simulates three of our soccer robots *Paderkicker* [9]. They have to learn how to optimally perform a given task under strict resource constraints.

2 Related Work

Many existing approaches have shown, that communication can and should be used in MAS to improve the group performance [10, 11]. Riley and Veloso [12] demonstrate how coaching between the individual agents can improve the performance for Q-learning agents. Tan [4] investigated the issue of exchanging information between multiple Q-learning agents. He found that the exchange of learned policies among agents "speeds up learning at the cost of communication". Dahl et al. [13] show how inter-agent communication in conjunction with

RL can improve the capability of agent groups to organize themselves spatio-temporally. In his work the agents communicate the reward to speed up the learning process — the state space itself is predefined and kept fixed for the whole learning process. In contrast to Dahl our approach deals with the adaptation of the state space based on the experience level of the communicating agents.

In this work we are not concerned with the optimization of joint actions in cooperative MAS, as it is investigated e.g. by Kapetanakis et al. [14] who demonstrate how agents employing their FMQ algorithm have the ability to converge towards the optimal joint action when teamed-up with one or more simple Q-learners which are in touch all the time. A more general investigation on cooperation in MAS is done by Claus et al. [15]. Instead, we are interested in mechanisms that allow for robust spreading of learned efforts between agents where the experience and the number and the visible agents may vary. The work on expert balancing algorithms, e.g. [16], typically relies on a fixed set of experts that issue recommendation at fixed time interval and the agent in question only has to choose one every time step. However, situations of this kind are very seldom in real-world application. Often we are happy if there is someone within reach to communicate with and we have to make ad-hoc decisions about how much value his information provides and how we integrate it into our own knowledge. For this situation we present our approach.

3 The Problem Domain

We consider an environment where mobile robots have to maximize their performance performing an abstracted task while keeping track of their limited resources. A robot collects a so-called task point in every time step when it executes the proper task action in the task area. The agent has to pay attention to resources of m types which are consumed at an individual rate. For each resource type the actual resource level is measured by a continuous value between 0 (resource of this type exhausted) and 1 (resource of this type is filled). The agent's main goal is to collect as many task points per lifespan as possible. For this it has to interrupt the main task in order to timely arrive at the filling station of the correct type that satisfies the agent's resource needs. In the environment there are multiple energy bases for every resource type. If one resource level is zero the agent dies and is restarted with zero task points. This way they have to trade off task accomplishment and lifetime extension through timely taking care of resources.

4 Architecture

The agents learn on two levels (Fig. 1): 1) the individual learning level, where the best mapping from the state space to the action space is learned using standard Q-learning, and 2) the knowledge exchange level, where the agents exchange their knowledge when encountering each other.

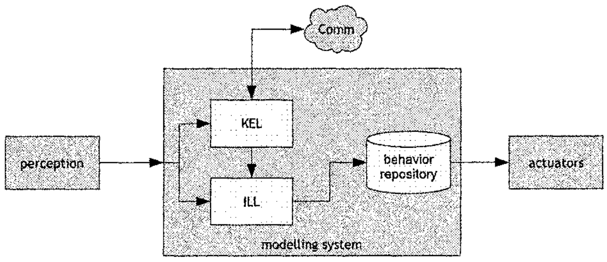

Fig. 1. Agent architecture: The interplay between the knowledge exchange level (KEL) and the individual learning level (ILL).

4.1 Individual Learning Level (ILL)

An agent is provided with hand-coded reactive behaviors [17] that move toward one of the two resource type areas or to the task area, respectively. The behaviors are constructed out of low level basic actions that move the agent toward the desired goal and avoid obstacles on the basis of potential fields [18]. The agents have to learn the best mapping of their intrinsic resource level state to the predefined behaviors using reinforcement learning. For every resource the following states are possible: 0 for "drive immediately to the proper energy filling station", 1 for "resource OK", and 2 for "resource maximum". State 2 is used to denote that the agent can stop the refuel process. States 1 and 2 say that it is save to perform the actual task to collect performance points. The threshold that the agent will have to adapt at runtime through cooperative learning divides states 0 and 1 and thus discretizes the continuous resource fill level into discrete states usable for RL. The reinforcement-learning agents use the one-step Q-learning algorithm [19] to learn the correct mapping of the resource input state vector to the proper behavior.

Since the actions take some time to be accomplished, the Q-values are not updated at every time step, but only after the chosen action has been finished. The Q-values for the state-action pairs are calculated with the standard Q-learning approach:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_x Q(s',x) - Q(s,a)\right) \tag{1}$$

Here, s denotes the state combining the state of two resource types, in this example called "R" for red and "B" for blue. The variable a denotes one of the three approaching abstract behaviors. The learning factor α was set to

$\alpha = \frac{1}{n(s,a)}$ with $n(s,a)$ being the counter for executing action a in state s. The probability $p(a|s)$ for choosing the best action a in state s is calculated as

$$p(\arg\max_a Q(s,a)|s) = 0.999^u \cdot 0.3\,,$$

with u being the global update counter that is increased at every Q-value update. The reward was given after $r = 0.01(c_w - c_b) + 5c_t$, with c_w counting the time steps the agent has been performing its task and c_b counting the time steps the agents resources were below the threshold. c_t was set to 1 if the agent managed to turn around one of its resources meaning a successful refuel since the last update, and 0 otherwise. The discount factor γ is set to 0.1. These values were empirically determined to run reasonably well even for a pure RL approach without the expert learning level for later comparison.

4.2 Knowledge Exchange Level (KEL)

In parallel to the learning process at the ILL the agents where enabled with communication means to optimize the segmentation of their continuous state space through group learning. The knowledge to exchange in our problem domain are the threshold values of each resource type in $R = \{1,\ldots,m\}$. These values determine when an agent will stop his task performing actions and drive immediately to the proper energy filling station. The adjustment of the thresholds is done at run-time, thereby adjusting to a moving target, since it might for instance become easier to reach a particular refilling station.

If an agent has a found a better segmentation this will be acknowledged at the next communication process, because it will most likely lead to a better performance. The knowledge weighting in combination with mutation is done as follows: The knowledge of agents that have a better recent performance will also have a greater impact on the final adapted knowledge of both agents. Furthermore, also the agents' knowledge accuracy should be dependent on the agents' experience. I.e. the less experienced an agent is the more noisy it's knowledge will enter the final outcome. This is done by trading off the impact and the mutation rate at the exchange of knowledge when communicating, leading to the computation of the adapted knowledge: The new thresholds of agent $i \in D$, where $D \subseteq A$ with $A = \{1,\ldots,n\}$ denoting the entire group of n agents depending on m different resources $r \in R$ is described by equation (2):

$$t_i^r = \sum_{a\in D} \frac{w_a}{\sum_{a\in D} w_a} \cdot \hat{t_a^r} \qquad (2)$$

$\hat{t_a^r}$ is drawn randomly from the Gaussian distribution $N\left(t_a^r, \sigma_a^2\right)$ with the standard deviation $\sigma_a \sim e(a)^{-1}$, thereby modeling the mutation that is introduced at every communication process. The impact w_a is proportionally dependent on the expert state $e(a)$ of that agent. $e(a)$ can be modeled to denote the moving average of the agent's lifetime or task achievement. The greater $e(a)$ is the more impact on the estimation of threshold t_i^r it has.

Thus, the "expert impact" $e(a)$ affects the overall threshold adjustment in two ways: On the one hand directly through the weight w_a, and on the other hand through the accuracy modeled by the normal distribution $N\left(t_a^r, \sigma_a^2\right)$. This has the counterpart in real life where usually more attention is paid to the experts than to newbies. In practice, the expert impact $e(a)$ of agent a defined as $e(a) = lifetime(a) + performance(a)$ has shown to yield a reasonably good expert measurement in our domain. The performance is calculated as one point per time step when performing the task. The thresholds are adjusted in two ways:

1. At a fixed time frame the agents get the chance to communicate with other agents staying close enough defined by a radius. Given, that agent i comes close enough to communicate with j, they both exchange their own estimation of the thresholds t^r, $r \in R$, as shown by Formula (2) from the i's point of view. In this case, $D = \{i, j\}$. The thresholds are computed separately for every agent which leads to different thresholds for both and encourages them to explore the state space. It showed that letting the agents communicate too often will prevent the convergence to sound threshold values since the agents have no time to see the effect of the new thresholds.
2. If an agent dies, because one of its resources is exhausted, it is restarted with full resources and its thresholds are calculated from the thresholds of all agents ($D = A$) randomized and weighted dependent on their experience $e(a)$.

In this experiment, we restricted the agents to apply the group learning to only the thresholds segmenting the state space. However, it could be easily applied to other areas like, e.g. the Q-values.

5 Experimental Setting

The approach has been evaluated in simulation with three Pioneer2DX robots [20] having sonar (90°) for obstacle avoidance, laser range-finders for detection of the individual markers, and differential gear. The experiments were performed on the *Player/Stage* [21] simulation environment. Usually, controllers written for the Stage simulator can be used on real Pioneers [13]. The agents were equipped with 360° fiducial finders, so that we could turn off foraging and map building control and concentrate on the learning task at hand.

The agents were anonymous in that they could not detect each others identity. When communicating they only were allowed to transmit their evaluation function, i.e. the resource thresholds for the two energy types "R" and "B", and their expert state, i.e. their current age. The distribution of the energy bases in the environment can be seen in Fig. 2. At the upper and lower left corners of the 8x6m area there are located two resources of type "R". In the middle of the left wall there is the task area: agents staying near this marker were said to execute the desired abstract task and thus gathering a point for every predefined

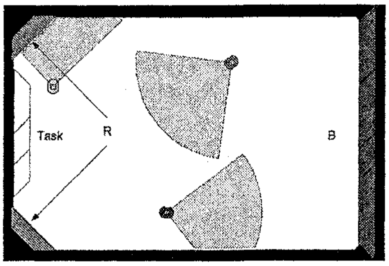

Fig. 2. Environment containing the task area and the energy bases of type "R" and "B".

timer interval. At the right side there is the resource of type "B" located. The setting thus has the property that resource "R" is located right beside the task area, whereas the agents have to travel across the entire field width to arrive at resource "B" to fuel their resource tank.

The environment shows the wanted realistic properties without distracting from the experiment under investigation: 1) It differs from the blocks-world examples, in that the individual actions take time that cannot be foreseen and perception being noisy. Although the distance between the different marker areas are approximately known and could be computed, the time to travel from one to another cannot be calculated in advance with adequate accuracy, because the interference of agents that have not managed to dodge each other introduces the time needed to perform a relieve maneuver. 2) Resources can be blocked by another agent so that an agent that also wants to reach the resource has to wait for an unknown time.

The parameters were set as follows: In formula (2) β was set to 0.2, $\sigma_i^2 = 0.2 \cdot e^{-\frac{e(i)}{100}}$, setting the variance to 0.2 for unexperienced agents and decreasing with the agent's experience. Although this also had to be empirically determined, it usually has to be done only once — changes in the environment do not render the once chosen value for σ useless, as opposed to the thresholds which have to be measured empirically every time anew without this form of social adaption.

6 Experimental Results

The experiment was run for 120 minutes per trial. We performed 10 trials for the random version, and 30 trials for the pure RL experiment (only ILL) and the hybrid version (ILL + KEL).

One would normally think that setting the threshold of "R" to something like 0.2 and that of "B" to approximately 0.8 would yield the best performance and lifetime of the agents, since "R" is much nearer to the task area and thus the behavior to drive to the "R" refuel station could be triggered much more later. However, as the experiments showed, the agents' thresholds converged to approx. 0.54 for "R" and 0.66 for the "B" thresholds as shown by Fig. 3. A close look to the experiment showed that it is not much more difficult for the agents to reach the "B" energy base as it is for the "R" base, although "B" is further away. This is the case because the area in front of "R" is most of the time occupied by another agent and thus unreachable for the agent, whereas "B" is farer away but most of the time clear of other agents since there are more "B" bases available.

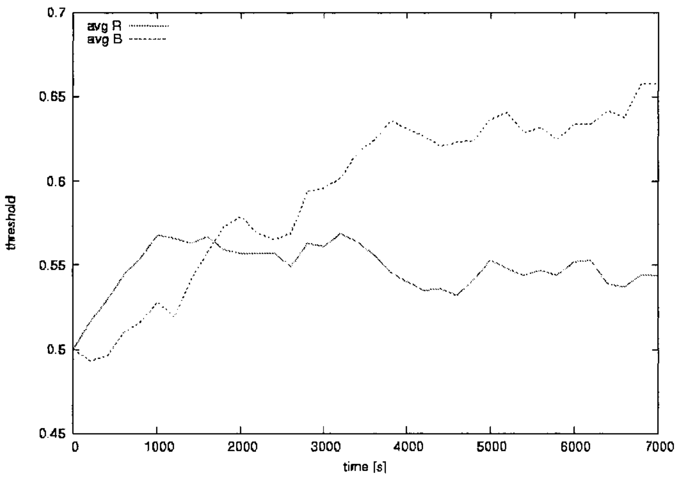

Fig. 3. Development of the threshold adjustment through group learning. The upper curve shows the threshold development for resource "R", the lower for "B"

The average lifetime development of the MAS is displayed in Fig. 4. We included the random policy for additional comparison. The significant data starts at $50s$ – the point when the first resource is exhausted. After approximately 100 seconds the agents start to die because they have not yet learned which action to take when the individual resources are going to be exhausted. As the agents learn more and more at the ILL the pure RL method departs from the random method, but as the graph suggests the parameters for the thresholds seem not to be optimal since it performs not that much better than the random method. The hybrid method, having RL at the policy level and group learning as described in section 4.2 at the KEL, outperforms the pure RL version. And this not only by the average lifetime measure, but also in the performance plot, as can be seen in Fig. 4. It takes, however, $3000s$ until the hybrid version performs better than the pure RL version. After $5000s$ the difference is significant and amounts to approx. 200 performance points at the end of the run.

One might ask why the hybrid solution does not reach a point where the agents live eternally, since they must have found the optimum thresholds. This

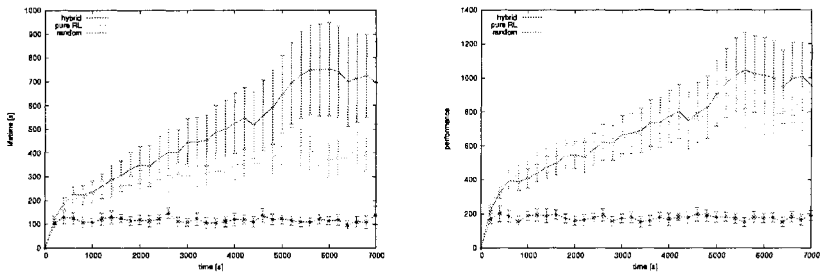

Fig. 4. The average lifetime (left) and performance (right) of the agents. Shown are the means and error bars using 95% confidence interval.

is because of the agent's interferences in front of the resource filling stations, which is not predictable, so that the death is always possible. The agents are thus trading off life time with performance points. Even with both resource thresholds set to 1.0 they would not be immune to death.

As can be seen in Fig. 4 the confidence intervals are very big. This is consistent with real human societies where the transmitted knowledge does not immediately result into a sudden and clearly defined benefit. Here, a much longer simulation interval would be needed to see the development over several days to months. This is planned in our future work.

7 Conclusion

For Multi-Agent environments without a central intelligence and with only sporadic communication possibilities for the individual participating agents we have presented a method to combine the individual learning process with a new way to integrate the learned knowledge of other participants. Based on the previous success of the agents (their "expert state") their knowledge is weighted and mutated. This results in a greater impact and accuracy of the knowledge of the better performing agents. Thus we trade off the accuracy of considered to be correct knowledge with the utility of introducing mutation for exploring knowledge for better performance. As shown in the experiment, it is interesting that this can lead to solutions that are clearly not intuitive to the engineer, instead lead to better behavior when seen with the agent's eyes.

We plan to apply this approach to our real world soccer robots: In addition to the adaptation of the state space this also includes e.g. the individual behaviors, the reward function and the learning rate α. Furthermore, we plan to choose the optimal level of aggressiveness in a soccer play as seen from their perspective: Being as aggressive as it is possible while avoiding to receive the red card would be a nice experiment to investigate with our soccer robots *Paderkicker* [9].

References

1. M. L. Littman L. P. Kaelbling and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.
2. R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, 1998.
3. M. J. Matarić. Reinforcement learning in the multi-robot domain. *Autonomous Robots*, 4(1):73–83, 1997.
4. M. Tan. Multi-agent reinforcement learning: Independent vs. cooperative learning. In M. N. Huhns and M. P. Singh, editors, *Readings in Agents*, pages 487–494. Morgan Kaufmann, San Francisco, CA, USA, 1997.
5. L. Kleinjohann W. Richert, B. Kleinjohann. Learning action sequences through imitation in behavior based architectures. In *Systems Aspects in Organic and Pervasive Computing – ARCS 2005*, number 3432 in LNCS, pages 93–107. Springer-Verlag Berlin, 14 - 17 March 2005.
6. A. Saskevic M. Koch, W. Richert. A self-optimization approach for hybrid planning and socially inspired agents. In *Second NASA GSFC/IEEE Workshop on Radical Agent Concepts*, NASA Goddard Space Flight Center Visitor's Center Greenbelt, MD, USA, 2005.
7. R. Dawkins. *The Selfish Gene.* Oxford University Press, Oxford, 1976.
8. S. Blackmore. *The Meme Machine.* Oxford University Press, 1999.
9. M. Koch B. Kleinjohann, W. Richert, P. Adelt, and S. Rose. Paderkicker. http://paderkicker.upb.de, 2006.
10. T. Balch and R. C.. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52, 1994.
11. M. Matarić. Learning to behave socially. In *SAB94: Proceedings of the third international conference on Simulation of adaptive behavior : from animals to animats 3*, pages 453–462, Cambridge, MA, USA, 1994. MIT Press.
12. P. Riley and M. Veloso. Coaching advice and adaptation. In B. Browning D. Polani, A. Bonarini and K. Yoshida, editors, *RoboCup-2003: The Sixth RoboCup Competitions and Conferences.* Springer Verlag, Berlin, 2004.
13. T. S. Dahl, M. J. Matarić, and G. S. Sukhatme. Adaptive spatio-temporal organization in groups of robots. In *IEEE/RSJ International Conference on Robotics and Intelligent Systems*, pages 1044 – 1049, Lausanne, Switzerland, Oct 2002.
14. S. Kapetanakis and D. Kudenko. Reinforcement learning of coordination in heterogeneous cooperative multi-agent systems. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1258–1259, Washington, DC, USA, 2004. IEEE Computer Society.
15. C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *AAAI/IAAI*, pages 746–752, 1998.
16. N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *J. ACM*, 44(3):427–485, 1997.
17. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation RA-2*, pages 14–23, 1986.
18. R. C. Arkin. *Behaviour-Based Robotics.* MIT Press, 1998.
19. C. J. C. H. Watkins and Dayan. *Q-learning.* 1992.
20. ActivMedia. URL for the Pioneer robot: http://www.activrobots.com, 2003.
21. B. P. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the International Conference on Advanced Robotics*, pages 317–323, Coimbra, Portugal, Jul 2003.