Learning Capabilities of Agents in Social Systems

Nguyen-Thinh Le and Lukas Märtin and Niels Pinkwart
Niedersächsische Technische Hochschule
Braunschweig, Clausthal-Zellerfeld, Hannover, Germany
{nguyen-thinh.le, niels.pinkwart}@tu-clausthal.de, maertin@ips.cs.tu-bs.de

Abstract

In a social computational system, there exist not only social interactions between software agents but also between humans and agents. Through interactions with humans, agents can acquire more knowledge, e.g., in problem solving. Usually, agents are hard-coded with anticipated abilities and their knowledge cannot evolve dynamically. In this paper, we propose a strategy-based approach to enable agents learning from humans in conflict situations. The learning process consists of four phases: 1) the conflict between a human and an agent is detected, 2) the human initiates a communication with the agent and proposes a strategy to solve the conflict, 3) the human's strategy is evaluated, and 4) the agent applies the most effective strategy in a new similar situation. The contribution of the paper is two-fold: it presents a new agent learning approach in the area of multi-agent learning and proposes a way of cooperation between humans and agents in a social computational system to evolve agents' abilities.

1. Introduction

In a social computational system, the social behavior of many types of interacting system participants is considered, e.g., humans, software agents. Usually, in such a system agents are usually hard-coded with limited abilities in order to perform certain tasks. One of the challenges for research in social computational systems is how abilities of agents can be enriched to adapt to social dynamics. The question being investigated in this paper is therefore how agents' abilities can evolve by learning from humans.

We focus on the ability of agents to learn from humans in situations, where resource conflicts occur. Given a conflict problem to be solved, humans may apply several strategies. Researchers suggested that experts have some kinds of knowledge about problem categories and associated solution schemas [11]. When an expert solves a problem, she will identify the problem characteristics by associating it with previously solved problems. The problem will be assigned to a solution schema which might be applied to solve problems of that type. Le and colleagues [8] defined the term *solution strategy* for any general domain as follows: "A solution strategy is based on the available means which can be used to deal with frequently occurring problem situations." For instance, in the domain of travel planning, if the task is to find a route between two places, possible strategies are e.g., driving by car or taking a train. In this paper, the term *strategy* is noted as a way of solving a problem using available means and this way is usually applied by domain experts for a certain class of problems. Under this assumption, we propose a strategy-based learning algorithm for agents. The algorithm consists of four phases. First, the human meets agents in a resource conflict situation. Then, in the second phase, the human initiates a conversation with the involving agents and propose a strategy to solve the conflict. In the third phase, the agents evaluate the effectiveness of the proposed strategy. The agents may learn several strategies from different humans. In the last phase, the agents apply on one of the most effective strategies they have learned in similar conflict situations.

2. State of the art

In the context of agent learning through interaction with humans, Kaiser et al. identified two classes of learning tasks [5]: 1) learning *for* communication and 2) learning *from* communication. While the first task of an agent is learning how to communicate with a human and to adapt the preferences of the human, the goal of the second task is to receive instructions from a human to solve problems in a certain situation. From the perspective of using communication between agents to improve learning, Sen and Weiss [13] classified learning tasks into two levels: low-level and high-level communication. The first type of communication covers simple query-and-response interactions for the purpose of exchanging missing information (e.g., knowledge/belief) to share information between agents. The second type of communication is characterized by more complex communicative interactions (e.g., negotiation, mutual explanation). The purpose of the high-level communication is to share understanding between agents by combining and processing information. The authors suggested that learning using high-level communication is a characteristic of human-human learning.

While a range of machine learning techniques, e.g., reinforcement learning, decision tree learning, has been applied successfully for the first class of learning tasks, for the second class and for high-level communication-based learning, successful learning techniques are rarely found in literature. This can be explained by the fact that learning through communication with humans and human-human learning are relative complex learning scenarios to be modeled in multi-agent systems. Thawonmas et al. [16] developed an approach to extract condition-action rules from a base of decision making behaviors of humans based on decision-tree techniques. The authors used a *RoboCup* simulation system to enable a human player plays soccer against two agents. Based on log data provided by the system, condition-action rules are derived and then applied to the agent. By this way, the agent adapts decision-making behaviors of the human player. The evaluation of the system showed that the agent can adapt almost human decision-making behaviors in a small scenario of playing soccer after five games between a human and agents. Taylor et al. (2011) proposed a human-agent transfer (*HAT*) approach which combines transfer learning, learning from demonstration and reinforcement to achieve fast learning. Following the *reinforcement learning* approach, an agent learns to take actions to maximize their utility which is accumulated through rewards [15]. Reinforcement learning techniques have been successfully applied in several applications (e.g., [12]), however, require a large amount of training data and high exploration time. *Learning from demonstration*, also referred to as *imitation learning* and *apprenticeship learning*, is a technique which aims at extending the capabilities of an agent without explicit programming the new tasks or behaviors for the agent to perform. Instead, an agent learns a policy from observing demonstrations [1]. Applying learning from demonstration techniques, agents learn directly from humans without explorations, and thus less time would be required than the reinforcement approach. However, the quality of demonstrations depends heavily on the abilities of the human teacher. Taylor et al. combined these both approaches and applied transfer learning to transfer knowledge form a human to an agent. The work reported that combining these three learning techniques results in better learning performance than applying each single one.

To enhance the learning ability of agents, six strategies for automated knowledge acquisition proposed for expert systems might be applied [9]: 1) Rote learning - the knowledge required to perform some tasks is incorporated into an expert system; 2) Learning from instruction or by being told - the learning system receives instructions from a teacher (e.g., a human), transforms this knowledge to its own representation, and integrates it with prior knowledge for effective application; 3) Learning by deduction - in addition to the task of translating input knowledge into an internal representation, the learning system carries out deductive inference to verify its truth of the acquired knowledge, to determine consequences of the knowledge, or to transform the acquired knowledge into more useful forms; 4) Learning by analogy - the learner attempts to transfer its existing knowledge applicable for one problem to another similar problem; 5) Learning from examples or concept acquisition - the learning system has to induce a general concept description from a given set of positive (and optionally negative) examples of a concept; and 6) Learning from observation and discovery - a learning system observes changes in an environment, creates classifications from given observations, forms general rules/theory to explain a given phenomenon. Michalski [9] suggested that if we know the procedure of problem solving precisely, then knowledge should be incorporated directly, i.e., applying rote learning or learning from instruction. In this case, teaching an agent by instruction is simpler and better than to engage it in an inductive learning process. In cases, where precise algorithms are unknown or difficult to construct, applying learning by analogy or inductive learning strategies (learning from examples, learning from observation and discovery) are most appropriate.

Since in a social computational system humans are participants of a social environment and may apply a wrong problem solving strategy, the approach of learning from instruction may result in agents acquiring wrong knowledge. Therefore, it is required to evaluate knowledge transferred from humans. Similarly, an inductive learning technique like learning from examples seems to be appropriate for agents learning from humans in conflict situations. However, when humans perform a procedure of problem solving, agents are not able to assess whether that procedure is useful or effective. It raises a need to evaluate the procedure of problem-solving after each human has performed it. We propose the learning by deduction approach for agents in a social computational system, whereas the effectiveness of each problem solving strategy performed by humans is rated by each agent with respect to its satisfaction. The rating for each problem solving strategy indicates how effective it has been used to solve a conflict situation and is also the means for agents to decide on the most useful strategy when they encounter a similar conflict situation.

3 Case study: Smart Airport

In order to illustrate the learning approach pursued in this paper, we consider an airport departure scenario as a representative for a social environment. The airport consists of static (single-lanes, two-way roads, entrances, check-in counters, gates, plane parking positions, and charging stations) and moving objects (humans, autonomous transportation vehicles (ATVs), and human-controlled vehicles (HCVs)) (see Figure 1).

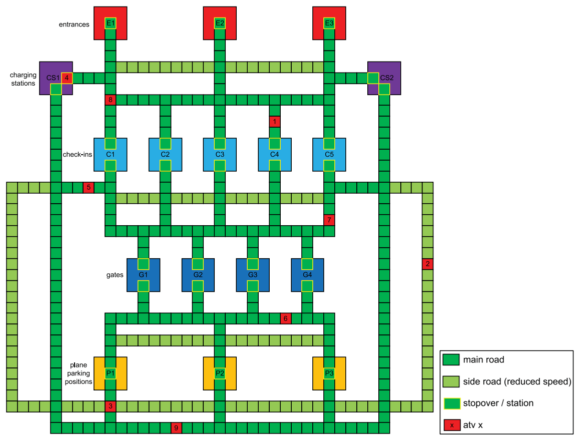

Figure 1. Road Grid of the Airport

Passengers can request a transportation service at an agency that provides vehicles (ATVs and HCVs) and manages passengers' orders. In a computational system, vehicles can be implemented by agents. An transportation order contains of start/end positions, pickup time, and latest time for drop-off. The start and end positions build a route, e.g., from an entrance to a check-in counter. Since both ATVs and HCVs need energy to move, they are equipped with batteries which need to recharge regularly at charging stations.

In this airport scenario, different types of conflicts might occur. We focus on resource conflicts, i.e., two or more participants compete for one resource. Typical resource conflict situations are:

1. At least two (max. four) vehicles are approaching a crossing. One of the vehicles needs the priority to pass through the crossing first. In this situation, the resource required by the vehicles is the crossing.
2. Several vehicles are running out of energy and need to be recharged, while the charging station might be occupied. The resource required by the vehicles is the charging station.
3. Vehicles have to take passengers to unoccupied check-in counters. The resource is a check-in counter. In reality, a check-in counter usually is occupied by one or two personnel, and thus it has a maximal capacity of two units.

4. Strategy-based learning from humans

Under the assumption that humans have a set of strategies for a certain conflict situation, we propose a strategy-based learning approach which consists of four phases:

Phase 1: Recognizing a conflict situation According to [17], a conflict is an opportunity for learning, because there occurs a social pressure to solve a conflict when two individuals disagree in a situation. Through resolving a conflict, individuals may change the viewpoint and their behaviors.

To detect resource conflicts, we apply a logic-based conflict model and the conflict detection mechanism. This conflict model assumes that an agent is able to see its peers within its limited view scope. Thus, the conflict detection mechanism makes use of the agent's *belief* about the world state within its scope. That is, each agent has information about the last, current, and next possible position of other participants existing in its scope. Based on this belief, an agent is able to identify other agents that will release/require a resource (an environment element) which is also required by itself. A *potential conflict* for an agent is defined formally as follows.

Definition 1 *Let A be an agent, its current position is $\langle X, Y \rangle$ and its next action is to require an environment element E at position $\langle X_{next}, Y_{next} \rangle$. Let α_E be the set of agents that are occupying E, $\alpha_{release,E}$ and $\alpha_{require,E}$ be the sets of agents (excluding A) that will release/require E, respectively, A has a* ***potential conflict****, denoted as $conflict(E, scope(A), \alpha_E, \alpha_{release,E}, \alpha_{require,E})$, iff $|\alpha_E| - |\alpha_{release,E}| + |\alpha_{require,E}| + 1 > C$, where C is the capacity of E and $scope(A)$ is the scope of A.*

Using Definition 1, an agent which intends to consume an environment element in the smart airport scenario, e.g., a crossing, a charging station, or a check-in counter, is able to detect potential conflicts.

Phase 2: Learning through communication Given a conflict situation C, there exists a set of possible strategies $\{S_1, .., S_n\}$ possibly applied by a human. A strategy is defined formally as follows:

Definition 2 *A strategy is a sequence of questions and answers $\{Q_1A_1, ..., Q_nA_n\}$, where questions Q_i are initiated by a* teacher *and answers A_i are carried out by a* learner*. A question is of one of the request types: performing an action, querying data, checking a predicate, or confirming an information.*

In our learning environment, a human plays the role of a *teacher* (a human) who sends a question to a *learner* (an agent) and the agent is in charged to answer human's questions. This way, the agent adapts the sequence of requests which have been performed by the human. This sequence of requests is applied for further similar conflict situations. In order to establish conversation between a human and an agent, a communication ontology needs to be defined.

For instance, when a HCV meets an ATV at a crossing, a potential conflict occurs as described in Section 3. In this conflict situation, the human may apply one of the following strategies: 1) calculating the priority based on the urgency of transportation tasks, 2) calculating priority based on energy states of the HCV and the ATV, or 3) the strategy of politeness, i.e., give way to the participant without requesting to calculate the priority. Applying one of these strategies, the human may initiate a conversation with the ATV as follows:

1. HCV $\rightarrow$ ATV: *Calculate priority* based on my *task*
2. ATV $\rightarrow$ HCV: My *priority is higher*
3. HCV $\rightarrow$ ATV: You *have way*
4. ATV $\rightarrow$ HCV: I confirm *OK*

In general, humans may use multi-modal interactions to indicate their strategy: e.g., using a common language, or non-communicative acts (gestures or movements). Inferring a humans' strategy from non-communicative acts is beyond the scope of this paper. In the approach pursued in this paper, depending on the sequence of requests initiated by the human to perform actions, the agent can derive which strategy the human currently intends. For instance, if the HCV requests the ATV to compare the priority of two tasks, than the strategy pursued by the HCV is comparing the urgency of the tasks. Peer-conversations are necessary to retain de-centralism of the system.

Phase 3: Evaluating human's strategies After the human has communicated with the agent, the conflict should have been solved, i.e., the resource can be allocated in a sequence according to the conversation between the participants. However, participants might not be delighted with the strategy proposed by the human. E.g., an ATV might have to give way to other participants, because it has lower priority in the context of comparing transportation tasks, but this ATV needs to be recharged as soon as possible because it's energy state is low. Thus, this raises the need to evaluate the effectiveness of the strategy proposed by the human in each conflict situation. For this purpose, each agent involved in the conflict situation has the opportunity to rate a proposed strategy. Let the rating scalar be the interval [0;N] where N is the best rating. The total rating for the strategy X which has been initiated by a human is $rating(X) = \sum R_A$, where $R_A \in$ [0;N] is the rating by an agent A involved in a conflict situation.

The total rating for each strategy applied by the human should be maintained and available for all agents. The agents involved in the same conflict situation need to share their ratings. Here, we have to make a trade-off between a centralized coordination and intensive peer-communication. Using peer-communication, each agent has to send its rating to its peers. However, this solution is very communication-intensive. An alternative is using a data base to maintain the strategies for different conflict situations and each agent updates the total rating for each strategy. In the approach followed in this paper, we choose the second option. Table 1 illustrates a partial data base of strategies for two conflict situations in the airport scenario: 1) *crossing*, where several ATVs/HCVs want to pass a crossing and 2) *charging station* where the energy of ATVs can be recharged. The third column of the table indicates the total rating of all strategies which have been applied by humans. According to that, for the crossing situation the strategy of calculating the priority based on transportation tasks has been evaluated by ATVs as most effective.

Table 1. Strategy Evaluation Table

Conflict	Strategy	Total Rating
Crossing	Politeness	0
Crossing	Task-based	10
Crossing	Energy-based	5
Charging station	Energy-based	15
...	...	...

Phase 4: Applying the best strategy When an agent detects a conflict with other agents in a situation in which it had a conflict with humans before, the agent takes the set of strategies which it has collected by learning from humans to apply. The best strategy of this set is determined by querying the rating in the strategy data base. The strategy which has the highest rating is taken the best one which can be used. Once again, after the agent has applied that strategy, its peers have the possibility to update the total rating in the strategy data base. E.g., Table 1 indicates that for the conflict situation *crossing* the task-based strategy is most appropriate, therefore, the ATV that has learned this strategy will initiate a conversation like in Phase 2.

A question arises that which agent should initiate a conversation in case of a conflict situation where no human involves. For this purpose, either an agent who has acquired knowledge should initiate the conversation or one of the involving agents is selected randomly.

5. Implementation

We implemented the airport scenario using the *JRep simulation platform* [4]. JRep is an integration of Repast Symphony and the JADE Framework. Repast provides a toolkit for visual simulations of multi-agent systems. JADE supports communication and interaction protocols according to FIPA-ACL. The environment of the airport is implemented in Repast, the behavior of autonomous agents is specified in JADE. JRep is extended to support our approach of strategy-based learning from humans. Humans controlling the HCVs can interact with ATVs using a graphical user interface (GUI). This way, ATVs learn from HCVs through conversations.

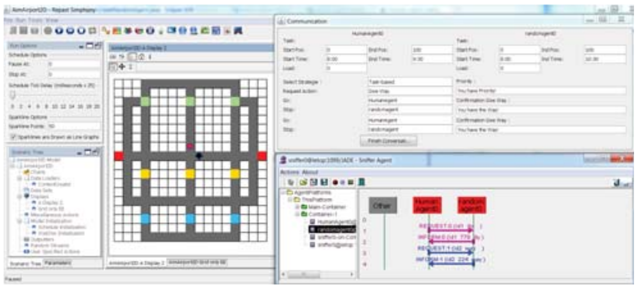

Figure 2. Enhanced Simulation Platform

As mentioned in the previous section, a communication ontology is necessary to enable conversations between agents/humans. FIPA-ACL[1] defines the elements of an ontology basically by *concepts*, *predicates* and *actions*. Our current implementation focuses on conversations at crossings in the road network of the airport. We define an *Airport-Ontology* for conversations between HCVs/ATVs and ATVs at the crossings with the mandatory elements in Table 2.

Table 2. Elements of the Airport-Ontology

Type	Element	Subelements
Concept	TASK	TASK_START_POSITION
		TASK_END_POSITION
		TASK_START_TIME
		TASK_END_TIME
	MOVING	MOVING_SPEED
		MOVING_DIRECTION
	POSITION	POSITION_X
		POSITION_Y
	PRIORITY	AGENT_YES_PRIORITY
Action	GIVE_PRIORITY_ACT	AGENT_GO
		AGENT_STOP
	CALCULATE_PRIORITY_ACT	SENDER_TASK
Predicate	IS_HIGHER	HIGH_PRIORITY_AGENT
		LOW_PRIORITY_AGENT

Figure 2 shows the simulation of the conflict situation at a crossing in a smart airport. On the left hand side of the figure, the airport environment is represented by a two-dimensional grid. The top part on the right hand-side of the figure illustrates a conversation between a HCV and an ATV. The GUI displays information about a transportation task, user-chosen strategy and requested actions of a HCV as well as the answers of the ATV. The bottom part on the right hand-side of the figure shows exchanges of messages between the HCV and the ATV controlled by Repast. By means of the GUI the user is able to directly control interactions of an HCA with ATVs.

6. Discussion

The approach of agents learning from humans presented in this paper is related to several research areas. First, the way of evaluating humans' strategies described in Section 4 (Phase 3) can be considered a filtering technique [14] which exploits the collaboration between agents to recommend the most effective strategy for a certain conflict situation. While a classical collaborative filtering technique uses users' rating data for items to infer recommendations based on calculating the similarity or weight between users or items e.g., Amazon shopping system, or GroupLens [7], our approach makes use of ratings of all agents to recommend the most effective strategy to solve a certain conflict situation.

Second, with respect to machine learning, the strategy-based learning process described in Section 4 is a supervised learning approach. Supervised learning is not widely used in multi-agent systems, because the interaction between agents is complex and this approach requires a critic/feedback that provides agents with correct problem solving behavior for a given situation. Nevertheless, there are several works in the context of mutual supervised learning e.g., [2, 3, 18]. Garland and Alterman [2] proposed a cooperative learning approach for heterogeneous agents which build their knowledge through their experiences. Each agent keeps planned or unplanned procedures which lead to successes in a case base. Goldman and Rosenschein [3] proposed a mutual learning approach in which each agent acts as the teacher of its partner. To acquire knowledge, the agents are trained by receiving examples and applying the concepts they have learned from their instructor. This approach avoids developing new a coordination of actions for similar problems. Williams [18] developed a learning algorithm for agents with different ontologies to share their knowledge in order to build a common ontology. This multi-agent learning approach has been deployed to assist groups of people in sharing knowledge. However, all these works did not deal with the issue of learning from humans.

Third, our strategy-based multi-agent learning approach can be considered as a type of cooperative multi-agent learning according to [10], i.e., a multi-agent system in which agents learn to cooperate with each other to solve a joint task or to maximize utility. The authors did an extensive survey of existing works in the area of cooperative learning in multi-agent systems. However, none of the systems reviewed in this survey considers learning from humans, most systems reported in this survey deal with the ability cooperation between agents to solve a joint task.

7. Conclusions and future work

In this paper, we have addressed the ability of software agents to learn from humans in a social computational system. The approach presented in the paper is based on strategy learning and consists of four phases: 1) a conflict between an agent and a human is detected, 2) the agent learns from the problem solving strategy proposed by the human

[1]FIPA-ACL, `http://www.fipa.org/specs/fipa00061`

through communication, 3) agents involving in the conflict situation evaluate the effectiveness of the strategy, and 4) the agent applies one of the learned strategies in a new similar conflict situation. We have simulated an airport as a social system where many types of participants interact with each other: e.g, autonomous vehicles, human-controlled vehicles, and passengers. Here, we have applied the strategy-based learning approach to autonomous vehicles which can have conflicts with human-controlled vehicles at crossings.

The learning approach presented in this paper contributes 1) to the multi-agent learning research area with a novel learning approach for agents and 2) to the social computational systems area a way of cooperation between humans and agents in order to evolve agents' abilities. There exist in literature many approaches for multi-agent learning and agents' cooperation. However, approaches to evolving agents' ability to learn from humans are seldom.

As forthcoming works, we will investigate our strategy-based learning approach in other conflict situations in the airport scenario, e.g., conflicts at charging stations or check-in counters. We believe that in these situations, humans also have a set of strategies to solve conflicts. Next, we will evaluate the strategy-based learning approach in this scenario. In addition, we intend to apply cognitive approaches (e.g., ACT-R [6]) to model each strategy as a sequence of actions such that agents can learn strategies in a human-like manner.

Acknowledgment

This work was funded by the NTH Focused School for IT Ecosystems. NTH (Niedersächsische Technische Hochschule) is a joint university of Technische Universität Braunschweig, Technische Universität Clausthal, and Leibniz Universität Hannover.

References

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Journal Robotics and Autonomous Systems*, 57:469–483, 2009.

[2] A. Garland and R. Alterman. Autonomous agents that learn to better coordinate. *Autonomous Agents and Multi-Agent Systems*, 8:267–301, May 2004.

[3] C. V. Goldman and J. S. Rosenschein. Mutually supervised learning in multiagent systems. In *Adaptation and Learning in Multi-Agent Systems*, pages 85–96. Springer-Verlag, 1995.

[4] J. Görmer, G. Homoceanu, C. Mumme, M. Huhn, and J. P. Müller. Jrep: Extending repast simphony for jade agent behavior components. In *Proc. of the 2011 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'11)*, 2011.

[5] M. Kaiser, V. Klingspor, and H. Friedrich. Human-agent interaction and machine learning. In *Proceedings of the 9th European Conference on Machine Learning*, pages 345–352, London, UK, 1997. Springer-Verlag.

[6] W. Kennedy and J. Trafton. Long-term symbolic learning in soar and act-r. In *Proceedings of the Seventh International Conference on Cognitive Modeling*, pages 166–171, 2006.

[7] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. GroupLens: applying collaborative filtering to usenet news. *Commun. ACM*, 40:77–87, 1997.

[8] N.-T. Le, W. Menzel, and N. Pinkwart. Considering ill-definedness of problems from the aspect of solution space. In H. W. Guesgen and R. C. Murray, editors, *Proceedings 23rd International Florida Artificial Intelligence Conference, FLAIRS-2010*, 2010.

[9] R. S. Michalski. *Learning strategies and automated knowledge acquisition: an overview*, pages 1–19. Springer-Verlag, London, UK, 1987.

[10] L. Panait and S. Luke. Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems*, 11:387–434, November 2005.

[11] N. Pennington and B. Grabowski. The tasks of programming. In J.-M. Hoc, T. R. G. Green, R. Samurcay, and D. J. Gilmore, editors, *Psychology of Programming*, pages 45–62. Academic Press Ltd., Newyork, 1990.

[12] M. Saggar, T. D'Silva, N. Kohl, and P. Stone. Autonomous learning of stable quadruped locomotion. In G. Lakemeyer, E. Sklar, D. Sorenti, and T. Takahashi, editors, *RoboCup-2006: Robot Soccer World Cup X*, volume 4434 of *Lecture Notes in Artificial Intelligence*, pages 98–109. Springer Verlag, Berlin, 2007.

[13] S. Sen and G. Weiss. Learning in multiagent systems. In G. Weiss, editor, *Multiagent systems*, pages 259–298. MIT Press, Cambridge, MA, USA, 1999.

[14] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advanve in Artificial Intelligence*, 2009:4:2–4:2, January 2009.

[15] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[16] R. Thawonmas, J. Hirayama, and F. Takeda. Learning from human decision-making behaviors - an application to robocup software agents. In T. Hendtlass and M. Ali, editors, *Proceedings of the 15th International Conference on Industrial and Engineering, Applications of Artificial Intelligence and Expert Systems*, volume 2358 of *LNCS*, pages 136–145. Springer, 2002.

[17] G. Wei and P. Dillenbourg. *What is 'multi' in multi-agent learning*, pages 64–80. Pergamon Press, Oxford, 1999.

[18] A. B. Williams. Learning to share meaning in a multi-agent system. *Autonomous Agents and Multi-Agent Systems*, 8:165–193, March 2004.