Collaborative Multiagent Learning: A Survey

Liviu A. Panait and Sean Luke

Department of Computer Science, George Mason University
4400 University Drive MSN 4A5, Fairfax, VA 22030, USA
lpanait,sean@cs.gmu.edu
http://www.cs.gmu.edu/∼eclab

Abstract. The field of Multiagent Systems is concerned with domains where several agents interact while solving competitive or cooperative tasks. Increasingly complex problem domains involving non-trivial numbers of agents revealed inherent difficulties with creating such systems. As such, the field witnessed a recent increased interest in Machine Learning techniques, capable of easing the programming efforts for approaching more challenging domains.

Despite the relative youth of the field, there are already several hundreds of papers trying to answer interesting questions in domains where learning affects the behavior of more than a single agent. Such questions target issues as varied as learning to communicate, modeling other agents in the environment, learning to compete or cooperate with the other agents, analysis of optimality for the learning algorithms, and many others. The large variety of papers generates a desirability for good surveys categorizing previous work.

We argue that there are two alternatives for learning in multiagent systems. A first direction is to have each individual agent learn how to improve its performance. The alternative is to have a single learning process that improves the behavior of the entire team of agents. Because of scalability problems with respect to increased numbers of agents, the majority of machine learning techniques can not be realistically applied to learn team behaviors. We discovered that most previous surveys concentrate on individual agents' learning, but neglect approaches at the team level.

In this survey, we propose a new arrangement for multiagent learning papers. As such, there is an entire category of papers dealing with learning behaviors for the entire team and issues associated with scalability to non-trivial numbers of agents, such as the heterogeneity of the team. A second class of papers is concerned with individual agent learning in multiagent domains. Based on their main research focus, the papers are categorized in sections dealing with optimality of learned behaviors, impact of locality of reward information, cooperation or competition relations among agents, and modeling other agents.

Additionally, we identified two issues relatively perpendicular to the team or individual levels of learning: problem decomposition and communication. The former is concerned with techniques for decomposing either the problem to be solved or the collective behavior into simpler independent components that can be more easily handled by the learning process.

An in-depth survey of the literature on the relationship between multiagent learning and communication revealed three approaches, each with its own particularities. The three such methods involve communication via either rapidly decaying information, slowly decaying information, or embodiment.

1 Introduction

There is an increasing interest in decentralized approaches to a wide range of complex real-world problems. Usually, approaches to such domains fall into the area of Distributed Systems, where a number of entities work together to cooperatively solve problems. The additional combination of the Artificial Intelligence (AI) and Distributed Systems areas adds scalability and adaptability properties to the solutions. This area of research is known as Distributed Artificial Intelligence (DAI).

Traditionally, Distributed Artificial Intelligence is divided into two classes of approaches. The first area, Distributed Problem Solving, is concerned with distributing the problem solving process. It usually involves a number of nodes that divide and share knowledge about the problem and the constructed solution. The second class of approaches is known as Multiagent Systems[1] (or MAS) and involves nodes with some degree of autonomy. We will concentrate on MAS approaches in this survey.

Learning is a central concept in Artificial Intelligence. An entire subfield of AI, namely Machine Learning (ML), is concerned with only studying learning techniques. The past fifteen years have witnessed increased research work in learning approaches for multiagent systems. The interest stems from fascinating questions on how multiple agents can learn to work together.

Despite the relative youth of the field, the number of multiagent learning papers is fairly large. This makes it very desirable for good organizations of the work. We discovered that most previous surveys neglect a large number of techniques and issues associated with learning in multiagent systems. The specific approaches consist of a single learning process for the behavior of the entire team of agents. This makes the techniques very similar to standard machine learning techniques, but additional questions related to the scalability of the approaches to large teams of agents are of special interest in the context of multiagent applications.

In this survey, we suggest yet another categorization for multiagent learning investigations. We believe that this taxonomy is a better organization that

[1] The is no consensus on the correct spelling of the name of the field: some researchers use *Multiagent Systems*, while others like to use a hyphen (*Multi-Agent Systems*) to emphasize the origin of the short form of the name (MAS). We feel it is about time for a single name to be accepted and used throughout the community. We opt for the first form of the name, Multiagent Systems, the same one used in many recent influential publications.

correctly encompasses a large number of papers and interesting issues usually associated with multiagent learning.

The main two categories of techniques are team and teammate learning. Team learning involves a single process that adjusts the behavior of the entire set of agents. This category of approaches is further decomposed into three subcategories based on the degree of heterogeneity imposed a priori on the team composition. As such, we distinguish between learning homogeneous and heterogeneous team behaviors. An additional new class of hybrid techniques combine the homogeneous and heterogeneous approaches in an attempt to further reduce the learning search space and to speed up the learning process.

The second category of techniques, teammate learning, contains approaches where each individual agent conducts its own learning process. We categorize work in this section based on the main focus of the investigation. Accordingly, we identifiy four directions of research. The first one includes papers that analyze the optimality of the learned behaviors in usually simple multiagent domains. Next, there is a set of papers investigating the impact of using different degrees of locality for the reinforcement received by individual agents. This issue is related to the credit assignment problem, and it appears to directly influence the emergent heterogeneity of the team and its final performance. A third class consists of papers that deal with issues related to the relation among agents in the multiagent learning setting. If in team learning the agents work together to improve the performance of the team, there is no predetermined cooperative setting in teammate learning. Rather, agents may be at times in competitive or cooperative situations among each other. At other times, their rewards may be completely uncorrelated, meaning that agents are not cooperating, nor competing. Further, the relations among agents may change over time, complicating further the learning task. A last class of papers investigate the task of creating models of the other agents.

Aside from the two main categories of multiagent learning, we investigate a number of issues that are common to both. We consider that dividing the work into subsections for the team and teammate learning might have apparently reduced the significance of the topics. Therefore we decided to allocate entire sections for the topics.

A first such topic is related to decomposing the learning task into simpler subcomponents that can be much easier solved by the learning process. This includes, among others, hierarchical learning approaches and shaping.

The second topic studies communication in relation to learning. Communication allows agents to share knowledge about the environment and their progresses toward individual or team goals. We identify three categories of communication and discuss issues associated with each of them. The first such category includes rapidly decaying information, the most widely used communication technique in the literature. A second type of communication uses slowly decaying information to convey the desired information for longer periods of time. The use of pheromones is an example of such slowly decaying communication mechanisms. The last category involves agents acting as brains associated with bodies (such

Agents	Interactions	Environments
number of agents	range	predictability
heterogeneity of team	bandwidth	richness of resources
complementarity of goals	frequency	episodicity
control architectures	persistence	discrete/continuous
roles in the team	(un)structured	
sensing abilities	signal/knowledge	
effective abilities	directory service	
representation abilities	variability	
	infrastructure	

Table 1. Degrees of variation for Multiagent Systems according to (Weiß, 1999; Huhns and Singh, 1998)

as in robotics applications), where the actual bodies can be used (by acquiring certain positions or by changing locations) to signal specific information.

The survey continues with a brief description of multiagent systems, followed by a presentation of multiagent learning and existing taxonomies for multiagent learning. The next sections describe team and teammate learning, with subsections for each of the major topics of interest. They are followed by sections on problem decomposition and communication in connection with multiagent learning. Later sections describe open research issues, and present information on commonly used problem domains and resources available in published form or on the internet. The paper ends with a set of conclusions, acknowledgments to the many people that helped create this document, and an extensive list of references.

2 Multiagent Systems

As the name implies, Multiagent Systems are concerned with distributed solutions involving multiple agents that interact to solve the problem. Unfortunately, the term agent has been and is currently used with a very large number of meanings. In this paper, we consider that agents are entities that exhibit a high degree of autonomy. Agents are capable of observing the current state of the environment, and can perform actions that change the state of the environment. According to Wooldridge and Jennings (1995), there are two general usages of the term agent:

- *Weak* agents exhibit , *social ability* (interact via some communication language), *reactivity* (perceive the environment and respond to it accordingly) and *pro-activeness* (plan for the future).
- *Strong* agents may have *beliefs, desires, intentions, knowledge, commitments* and other human characteristics.

Work in Multiagent Systems differs along several directions in terms of types of agents, types of interactions and types of environments. A collection of degrees of variation taken from (Weiß, 1999; Huhns and Singh, 1998) is presented in Table 1. Despite this large variation in characteristics, Jennings et al. (1998) suggest

that most Multiagent Systems applications have a set of common settings: *individual agents have incomplete information about the environment*, *there is NO centralized system control*, *information is decentralized and distributed throughout the environment*, and *computation is asynchronous*.

An important issue in Multiagent Systems is the possibility of agents to communicate with each other. Communication consists of modifying the state of the environment such a way that other agents can perceive the modification and decode the information. Agents can use communication to share information about the environment or about their goals and means to achieve them.

Depending on their interest, several authors provided different taxonomies for MAS applications. For example, Dudek *et al* classify swarm robotics applications in terms of *teamsize, range, topology and bandwidth for communication, team composition and reconfigurability*, and *the processing ability of individual agents*. In a collection describing application of Distributed Artificial Intelligence in the industry, Van Dyke Parunak (1996) differentiates between *agent characteristics* (heterogeneity of team, control architectures, input/output abilities) and *system characteristics* (for example, communication settings). Stone (1998); Stone and Veloso (2000) explicitly distinguish between *homogeneous non-communicating*, *homogeneous-communicating*, *heterogeneous non-communicating* and *heterogeneous communicating teams*, and also presents issues associated with each of the three categories. The authors make the interesting observation that a MAS system with communicating agents where the communication has unlimited range and bandwidth is equivalent to a centralized system.

3 Multiagent Learning

We define multiagent learning broadly: it is the application of machine learning to problems involving multiple agents. We think that there are two features of multiagent learning which . First, because multiagent learning deals with problem domains involving multiple agents, the search space involved can be unusually large; and due to the interaction of those agents, small changes in learned behaviors can often result in unusual, often unpredictable changes in the resulting macro-level ("emergent") properties of the multiagent group as a whole. Second, multiagent learning may involve *multiple learners*, each learning and adapting in the context of others; this introduces game-theoretic issues to the learning process which are not yet well understood.

This survey is mainly concerned with collaborative multiagent learning, rather than competitive learning methods. However, it is worth mentioning one important class of competitive multiagent learning problem domains: learning to play games. In fact, one of the earliest, and still celebrated, machine learning papers is concerned with learning to play checkers (Samuel, 1959). However, the bulk of learned game-playing work has been relatively recent. In particular, evolutionary computation has been successfully used to learn good performing players in competitive domains. For example, Luke (1998) learned soccer-playing softbot teams, and Fogel (2001) evolved a highly human-competitive checkers program called Blondie24. Evolutionary computation was also used to find game-players for Tic-Tac-Toe (Angeline and Pollack, 1993), Backgammon (Pollack et al., 1997; Pollack and Blair, 1998), Mancala (Davis and Kendall, 2002), Othello (Smith and Gray, 1993), pursuit-evasion (Cliff and Miller, 1995; Harvey et al., 1997), Go (Lubberts and Miikkulainen, 2001), Chess (Kendall and Whitwell, 2001), Poker (Kendall and Willdig, 2001) and Tag (Reynolds, 1994).

Another large class of learning in multiagent systems that we will ignore in this survey includes cases where a single agent learns while the other agents' behaviors are fixed. One of the many examples of such learning investigations is presented in (Grefenstette, 1991). This is single-agent learning: there is only one learner, and the behaviors are plugged into only one agent, rather than distributed into multiple agents.

The survey will discuss a number of issues in multiagent learning, but we introduce some themes which will recur several times during the survey and deserve some mention up-front:

Credit Assignment. When evaluating an agent, an important problem that needs to be solved is that of deciding which one of a series of actions the agent performed led to the reward received from the environment. Additionally, when the reward is attributed to a team's behavior, assessing the credit an agent receives, and its due part of the team reward, can also be very important to the learning process. The two subproblems are known as the *intra-agent* (deciding among an agent's actions) and *extra-agent* (deciding among a team's agents) credit assignment problems (Weiß and Sen, 1996).

Game-theoretic Problems: Competition vs. Cooperation. An important related issue is how to cast a given multi-agent problem as a cooperative problem (as opposed to a competitive one, or one with other game-theoretic features). This is not a trivial problem: many multiagent learning problems can exhibit unexpected interactions between agents as they gravitate towards equilibium with one another. Here we will approach the issue only by defining cooperation, competition, etc. in a relatively extreme fashion. If increasing the reward received by one agent leads to increasing the reward for another agent, we say that they are cooperating. If increasing an agent's reward leads to decreasing the reward received by another agent, we consider that the two are competing. If there is *no relationship* between the two (they are completely independent), we say that they are one another.

Co-adaptation. Another interesting, game-theoretic multiagent learning issue concerns the fact that the learning processes are not independent, but they affect each other. Consider: at some point in time, an agent observes the environment (containing other agents as well) and tries to improve its performance. This leads to a modification in its behavior. This modification is then sensed by the other agents, who change their behaviors in order to improve their performances as well. This "moves the goalposts" on the original agent: its newly-learned behavior may no longer be appropriate. Thus as the agents co-adapt to one another, the agent environment is essentially changing beneath their feet. Learning in the face of this dynamic is not easy: such co-adaptation can result in cyclical or chaotic adaptive behavior, or to gravitation towards balanced equilibrium rather than an optimum.

3.1 The Survey Layout

We have divided cooperative multiagent learning papers into two broad categories. The first category applies a single learning process to improve the performance of the entire team of agents. We call this category *Team Learning* because the learning process adjusts the behavior of the entire team as a whole. The other category applies an individual learning process , while still assessing the quality of the agents as a team. We term this approach *Teammate Learning*.

Research in Team Learning has broken down along different lines than that of Teammate Learning, primarily because of differences in the dynamics of the techniques. Accordingly we subdivide the research in each category along these lines. Team Learning has largely focused on issues of agent homgeneity versus heterogeneity and various hybrid methods. Teammate learning instead has focused on game-theoretic issues such as co-adaptation, credit assignment, cooperation versus competition, and modeling other agents.

We then discuss specific issues common across multiagent problems: methods of performing *task decomposition* (); the effect of inter-agent *communication* on the learning process, and approaches to performing communication; and the challenges of scalability and adaptive dynamics.

Other surveys have broken the field down in other ways than we have chosen here. Stone and Veloso categorize multiagent work only along two dimensions: *team heterogeneity* and *presence of communication* (Stone and Veloso, 2000; Stone, 1998). They use this taxonomy to discuss issues associated with learning approaches for the four categories of systems: homogeneous non-communicating, heterogeneous non-communicating, homogeneous communicating and heterogeneous communicating. Weiß (1997, 1999) uses a taxonomy based on the main focus of the research at hand. Weiss categorizes multiagent learning research into approaches dealing with *learning to cooperate and compete*, *modeling other agents* and *learning and communications*.

4 Team Learning

In team learning, there is a single learner involved: but this learner is discovering a set of behaviors for a team of agents, rather than a single agent. While this lacks the game-theoretic aspect of multiple learners, we argue that team learning is interesting in that because the team of agents interact with one another, the joint behavior arising from their interactions is often unexpectedly complex: this notion is often dubbed the *emergent complexity* of the multiagent system.

Team learning is an easy approach to multiagent learning because it can use standard single-agent machine learning techniques: there is a single entity that performs the learning process. This sidesteps the difficulties arising from the co-adaptation of several learners that we will later encounter in teammate learning approaches. Another advantage of a single learner is that the agents tend to act to maximize the team reward rather than their individual rewards. This makes agents behave altruistically rather than greedily. As we will see later, selfishness creates significant problems in teammate learning.

With a single learner, the issue of credit assignment among the agents may generally be ignored: it is often reasonable simply to divvy up credit evenly throughout the team. However in some situations it can be helpful to assign credit in order to determine which parts of the team need the most improvement[2].

Team learning has some disadvantages as well. A major problem with team learning is the increasingly (usually exponentially) larger state space for the learning process. For example, if agent A can be in any of 100 states and agent B can be in any of another 100 states, the team formed from the two agents can be in as many as $10,000$ states. This explosion in the state space size can be overwhelming for learning methods that explore the space of state utilities (such as reinforcement learning), but it may not as drastically affect techniques that explore the space of behaviors (such as evolutionary computation) (Salustowicz et al., 1997, 1998; Sen and Sekaran, 1996).

A second disadvantage is the centralization of the learning algorithm: all resources need to be available in the single place where all computation is performed. This can be burdensome in domains where data is inherently distributed.

Among the several chaMLmachine learning techniques, evolutionary computation seems particularly apropos to team learning. That is because EC is a sparse learning method: it does not build up a full model of the learning problem. Thus EC scales linearly in memory requirements with each additional agent.

Team learning may be divided into two broad categories: *homogeneous* and *heterogeneous* team learning. Homogeneous learners develop a single agent behavior which is used by every agent on the team. Heterogeneous team learners

[2] It can be argued that assigning credit is essentially applying different quality assessments to each agent, and if the learner tries to locally improve each agent based on its quality assessment, this is equivalent to a multiple-learner (teammate learning) situation. We have chosen to put such gray-area models in the team learning category.

can develop a unique behavior for each agent. Heterogeneous learners must cope with a larger search space, but hold the promise of better solutions through agent specialization. There exist approaches in the middle-ground between these two categories: for example, dividing the team into squads, with squadmates sharing the same behavior. We will refer to these as *hybrid* team learning methods.

4.1 Homogeneous Team Learning

In homogeneous team learning, all agents use the same learned behavior. Because all agents have the same behavior, the search space for the learning process is drastically reduced. The appropriateness of homogeneous learning depends on the problem: some problems do not require agent specialization to achieve good performance. For other problem domains, particularly ones with very large numbers of agents ("swarms"), the joint behavior search space is simply too large for heterogeneous learning, even if heterogeny would ultimately yield the best results.

Much of homogeneous team learning concerns itself with communication issues. We discuss such literature in Section 7, and particularly in Section 7.3.

Burke et al. (2002) applies homogeneous genetic programming to the N-prisoners puzzle. The problem consists of N agents, each assigned a boolean value. Each agent does not know its own value, but it knows the values of all other agents. The agents must figure out what boolean values they have been assigned without communicating with each other. Each agent can state its believed boolean value to be *true* or *false*, or it can pass. All agents act simultaneously. They all receive no reward if any of them states an erroneous value, or if all pass; otherwise, they all receive some positive reward. Burke *et al* show that genetic programming can learn non-trivial homogeneous strategies, but ones that are inferior to the best known solutions.

Haynes et al. (1995b,a,c); Haynes and Sen (1995); Haynes et al. (1996) present a series of results obtained by evolving behaviors for the predator-prey pursuit domain. When using fixed (random or greedy) algorithms for the prey behavior, the papers report results competitive to the best human-coded greedy algorithms, both with and without using information on the position of the other predators. However, when coevolving the prey and predator behaviors, the genetic programming system employed discovers a strategy for the prey that evades all previously reported hand-coded, greedy, and evolved strategies. The authors suggest that improved performances may be obtainable with communicating agents. Jim and Giles (2000) follow this direction and allow a genetic algorithm system to additionally evolve a communication language. The authors experiment with increasingly complex language constraints, and report that the evolved communicating agents exhibit performances superior to all previously reported work in this domain.

Quinn et al. (2002) investigate the use of evolutionary computation techniques for a team formation problem. Three agents start from random positions at relatively small distances apart (each agent can sense the others with its sensors). They are required to move the team centroid a specific distance while

avoiding collisions and remaining within sensor range. Quinn *et al* investigate the roles of team members by removing the agents one at a time. They conclude that the rear agent is essential to sustain locomotion, but it is not essential to the other two agents' ability to maintain formation. The middle agent is essential to keep the two others within sensor range, and the front agent is crucial to team formation. Therefore, even though the agents are homogeneous, they specialize (based on their relative positions) to better perform as a team.

Bassett and De Jong (2000); Bassett (2002) investigates the evolution of homogeneous teams for a cooperative surveillance task. The authors treat the work with homogeneous agents as a first step towards the more challenging evolution of heterogeneous teams. Their results show good performance, but the authors mention expecting improvements when using heterogeneous teams or coevolution in future work.

Salustowicz et al. (1997, 1998) compare PIPE and CO-PIPE (population-based algorithms similar evolutionary computation) and Q-learning in a simulated soccer domain. The results show that Q-learning has serious learning problems, attributed by the authors to the algorithm's need to search over all state utilities. On the other hand, both PIPE and CO-PIPE search directly in the behavior space and show good performance. The authors conclude that searching in behavior space may be preferable in other multiagent domains. A similar result is reported in (Sen and Sekaran, 1996), where a "Bucket-Brigade" evolutionary algorithm proves competitive with Q-learning in a coordinated navigation problem. An opposing result is reported in (Wiering et al., 1999), where a modified Q-learning outperforms the methods earlier reported in (Salustowicz et al., 1997, 1998).

Cellular Automata Arguably, cellular automata (CA) is a notable and oft-neglected problem domain for homogeneous team learning. A CA a field (a row or grid) of agents, each with its own internal state, plus a homogenous state-update agent behavior (the *rule*) applied by all the agents synchronously. This rule is usually based on the current states of an agent's nearby neighbors. CAs have many of the hallmarks of a multiagent systems: iteractions and communications are local, and behaviors are performed independently. A good survey of existing work in learning cellular automata rules is presented in (Mitchell et al., 1996).

One common CA problem, the Majority Classification[3] task, asks whether there exists an update rule which can — given any arbitrary initial configuration of agents each with an internal state of 1 or 0 — correctly classify the initial configuration as having more 1's than 0's or more 0's than 1's. This is done by repeatedly applying the update rule for some N iterations; if the agents have converged to all 1's, the rule is said to have classified the initial configuration as majority-1's. If it has converged to all 0's, the rule has classified the initial configuration as majority-0's. If it has not converged, the rule has not classified

[3] Also known as Density Classification.

the initial configuration. It is known that such a rule does not exist: the goal is to discover a rule which classifies most configurations correctly.

Mitchell et al. (1994) identifies four epochs of innovation for genetic algorithms when applied to the Majority Classification task. The first two epochs consist of correctly converging when the majority state is clearly distinguishable by a wide margin. In the last two epochs, the learning process discovers improved collective behaviors that can additionally expand blocks of consecutive, almost identical states.

Researchers have shown that coevolutionary methods can provide a better gradient for learning the Majority Classification problem (Werfel et al., 2000; Pagie and Mitchell, 2001; Juille and Pollack, 1998). In particular, Juille and Pollack (1998) present a set of rules with classification accuracy of around 86.3%. The best previous results was learned using genetic programming and has accuracy 82.326% (Andre et al., 1996). In comparison, the best human hand-coded result was proposed by Das et al. (1994) and has accuracy 82.178%.

4.2 Heterogeneous Team Learning

In Heterogeneous Team Learning, the team is composed of agents with different behaviors, and the single learning process aims to improve the performance of the entire team. This approach allows for more diversity in the team at the cost of increasing the search space. An interesting question is if this tradeoff pays off within the limited computation time allowed[4]. Further answers to comparing homogeneous and heterogeneous team learning are reported in the next section.

Quinn (2001a) reports improved results when sampling the team members from a population of individuals compared to cloning behaviors to obtain completely homogeneous teams. Further investigations suggest that heterogeneity is not always desirable, and homogeneous teams can have superior performances in domains such as foraging. Balch (1998) suggests that domains where single agents can perform well are particularly suited for homogeneous learning, while other domains that require task specialization are more suitable for heterogeneous approaches. His results are confirmed by Bongard (2000), who hypothesizes that heterogeneity may be a better performant on inherently decomposable domains.

Potter et al. (2001) suggest that domain difficulty may not be the major factor that requires a heterogeneous approach. They experiment with increasingly difficult versions of a multiagent herding domain obtained by adding predators. Potter *et al* show that increasing the number of skills required to solve the domain (adding a predator requires both herding and defending strategies) leads to significantly better results obtained when heterogeneous teams are used.

Andre and Teller (1999) applies genetic programming to learning a team of soccer playing agents for the RoboCup simulator. The individuals encode

[4] Because homogeneous teams are particular instances of heterogeneous ones, we expect a better performance if enough computation time is allowed. However, the question is whether better teams can be obtained in a limited amount of time in the larger search space.

eleven different behaviors (one for each player). Andre and Teller mention that the crossover operator (between teams of agents) exchanged "most of the time" genetic material between equivalent players (genetic material from player 8 in one team was exchanged with genetic material from player 8 in the other team). This is somewhat similar to the restricted breeding studied by Luke and Spector (1996).

Luke and Spector (1996) investigate possible alternatives for evolving teams of agents. In the experiments, Luke and Spector compare homogeneous, heterogeneous with restricted breeding (breeding enforced only among behaviors for the same agent) and heterogeneous with no breeding restrictions. The authors suggest that the restricted interbreeding works better than free interbreeding for heterogeneous teams, which may imply that the specialization allowed by the heterogeneous team representation conflicts with the inter-agent genotype mixture allowed by the free interbreeding.

Haynes and Sen (1996b, 1997b,a) investigate the evolution of homogeneous and heterogeneous teams for the predator-prey pursuit domain. For heterogeneous teams, each predator in the team has a strategy represented in each individual in the population. The authors present several crossover operators that may help the appearance of specialists within the teams to some extents. The results indicate that team heterogeneity can significantly help despite the apparent domain homogeneity. The authors suggest that this may be due to deadlocks generated by identical behaviors of homogeneous agents when positioned in the same quadrants (when receiving identical inputs, homogeneous agents will behave identically). Harvey *et al* report that the most successful crossover operator (TeamUniform) allows arbitrary crossover operations between strategies for different agents, a result contrasting the one reported in (Luke and Spector, 1996).

Another interesting approach to evolving team behaviors is presented by Bull and Holland (1997), where team agents are sampled from an evolutionary computation population of individuals. This leads to heterogeneous teams that perform well in their investigation.

4.3 Hybrid Team Learning

Luke (1998); Luke et al. (1997) report on a combination of homogeneous and heterogeneous approaches to team learning. Their work concentrates on evolving soccer teams for the Robocup competition, and they mention that the limited amount of time available before the competition diminished the probability of obtaining good heterogeneous teams. Instead, they compare the fully homogeneous results with a hybrid combination that divides the team into six squads of one or two agents and then evolves six separate behaviors (all represented in an individual's genome). The authors report that homogeneous teams performed better than the hybrid approach, but mention that the latter exhibited initial offensive-defensive squad specialization and suggest that hybrid teams might have outperformed the homogeneous ones if more time was allowed.

Hara and Nagao (1999) present an innovative method for hybrid group learning. Acknowledging the superiority of heterogeneous teams, but admitting the

problem of increasingly larger search spaces when many agents are present, the authors suggest an automated grouping technique. Their method is called Automatically Defined Groups (ADG), and in this paper it is successfully applied to a simple load Transportation problem and a modified Tile-World domain. In ADG, the team of agents is composed of several groups of homogeneous agents (similar to (Luke, 1998; Luke et al., 1997)); however, ADG allows for automatically discovering the right number of groups. Additionally, the acquired group structure may give insights into the cooperative behavior that solves the problem.

Miconi (2001) presents a variation where all agents have a behavior represented as an individual in the evolutionary computation system. The population represents in fact the entire set of behaviors for the team members. Local selection and mating, combined with a form of elitism, result in an interesting evolutionary learning algorithm. Miconi mentions that the algorithm allowed for the appearance of subspecies whose sizes varied depending on the content of the population. This led to an emergent formation of squads of different sizes, similarly to the work of Hara and Nagao (1999).

5 Teammate Learning

The most common alternative to team learning in cooperative multi-agent systems is *teammate learning*, where each agent on the team independently learns how to improve its performance and the performance of the team as a whole[5]. While team learning decentralizes the agents while they are acting in the environment, teammate learning further decentralizes the agents' learning procedures as well. Some research has presented domains where teammate learning outperforms both homogeneous and heterogeneous team learning (Iba, 1996, 1998). However, other investigations suggest that team learning may be preferable in certain situations (Miconi, 2003).

There are many teammate learning papers in reinforcement learning, but the notion is of some special interest to evolutionary computation, because of its close relationship with *coevolution*. In EC, coevolution divides the population into multiple subpopulations (separate evolutionary computation learning systems). The quality of agents drawn from those subpopulations are assessed by testing them together. For example, *competitive coevolution* might pit robotic agents against each other in a game of capture-the-flag. *Cooperative coevolution,* popularized by Potter et al. (1995); Potter and De Jong (1994); Potter (1997); Potter and De Jong (2000), treats each subpopulation as a source for agents which must work together in a team: for example, a cooperative coevolutionary

[5] Another possibility is to have some degree of partial decentralization, whereby joint behaviors for *subgroups of agents* ("squads') are independently learned. By grouping learned behaviors by squad, the behaviors a team of M agents can be learned by some $N < M$ learners. This approach may have the advantage of separately learning relatively uncorrelated behaviors, while still jointly learning the highly correlated ones. However, we are not aware of any paper investigating this approach.

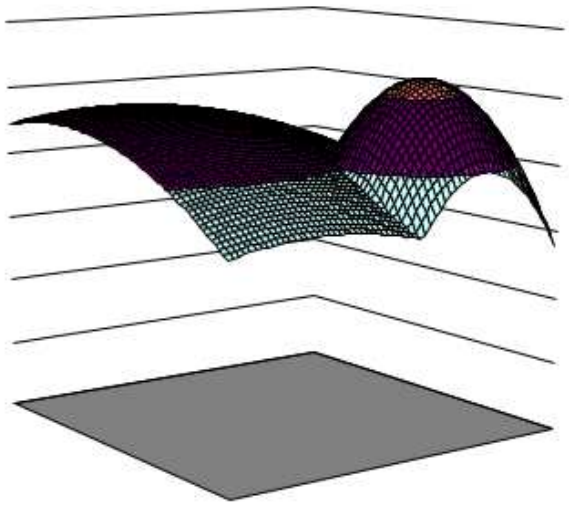

Fig. 1. Joint reward information for the actions pairs of two agents. The 64 actions of each agents is represented on the X and Y axes. The joint reward is represented on the Z axis.

system might evolve two separate subpopulations, drawing an agent from each to be tested together in a joint two-agent cooperative block-pushing exercise.

The primary advantage of teammate learning is that it explicitly breaks the large joint team search space into separate, smaller, individual search spaces. If the problem can be decomposed such that individual agent behaviors are relatively disjoint, then this can result in a dramatic savings in computational complexity. Another related advantage is that breaking the learning process into smaller chunks permits more flexibility in the use of computational resources to learn each process because they may, at least partly, be learned independently of one another. On the other hand, an advantage of team learning is that the system need only be concerned with a single learner, simplifying the design of the learning process itself.

A third advantage is the reduction in the search spaces for the learning tasks. Remember that in team learning of heterogeneous teams, the learning system had to learn a strategy for each of the individual agents. On the other hand, team learning has each learning process to only search in the space of behaviors for a single agent. Unfortunately, the reduction in the search space is accompanied by usually accompanied by a degradation in the quality of information that each of the agents receives as reinforcement.

The central challenge to teammate learning is that with multiple learners, each learner is adapting its behaviors in the context of other co-adapting agents over which it has no control. As the agents learn, they modify their behaviors,

Fig. 2. Reward information for the actions of one of the agents in the Two Peaks domain described in Figure 1. Each action is paired with the **ideal** action for the other agent.

Fig. 3. Reward information for the actions of one of the agents in the Two Peaks domain described in Figure 1. Each action is paired with a set of five random actions (uniformly distributed) for the other player, and the **maximum** of the joint rewards is plotted. The process is repeated three times, with one curve for each try.

which in turn can ruin other agents' learned behaviors by making obsolete the information collected by other the agents during earlier explorations. Agents may then need to re-explore the environment again and change their behaviors in this new agent context, but as soon as they do so, their new learned behaviors may yet again make obsolete each other's learned assumptions (and so on). The mutual co-adaption inherent in teammate learning presents evolutionary game-theoretic dynamics which are relatively new to the machine learning field; and there is so far relatively little agreement on how to deal with the problem, especially under in multi-agent environments with little or no communication.

This challenge manifests itself in a number of interesting problems. One such problem with teammate learning is keeping the agents focused on the performance of the team, rather then on their individual performances. This issue can be cast in game theoretic terms: a multi-agent learning system can have multiple Nash equilibrium points, and once the set of agents reaches one of the equilibria, no single agent has motivation to change its behaviors "for the good of the team". Only two or more agents simultaneously changing to a new behavior can get the team out of the equilibrium point. This creates a problem whenever the equilibrium point is a suboptimal team behavior.

Fig. 4. Reward information for the actions of one of the agents in the Two Peaks domain described in Figure 1. Each action is paired with a set of five random actions (uniformly distributed) for the other player, and the **average** of the joint rewards is plotted. The process is repeated three times, with one curve for each try.

Fig. 5. Reward information for the actions of one of the agents in the Two Peaks domain described in Figure 1. Each action is paired with a set of five random actions (normally distributed with mean 16 (the lower wider peak) and standard deviation 10) for the other player, and the **maximum** of the joint rewards is plotted. The process is repeated three times, with one curve for each try.

Another problem: when centralized performance measures are used, how can the learning system assess the contribution of each agent to the joint measure? This credit assignment problem may be sometimes very difficult to correctly solve or approximate. In some cases, equal shares of payoff are assigned to each agent. This may slow down the learning process because "lazy" agents receive rewards primarily due to other agents' sustained efforts, and so have little incentive to change their ways. On the other hand, equal shares of global reward can keep the agents focused on team (rather than individual) performance. The other extreme is to locally assess each agent's performance based on its individual behavior. This can lead to local, greedy behavior rather than globally optimal behavior. Several papers, discussed later, investigate the tradeoffs between these the two approaches and middle-ground methods which combine them.

A related concern with teammate learning is that "all-star teams" composed of best behaviors learned for each of the individual agents may not perform well together, because those best behaviors were learned in the context of specific other agents which are now not on the all-star team. For example, (Gordin et al., 1997) considers the situation where two agents must negotiate for resources; an optimal situation is where agent A allows agent B to have all the resources (or conversely, agent B allows agent A to do so). But forming an all-star team out

Fig. 6. Reward information for the actions of one of the agents in the Two Peaks domain described in Figure 1. Each action is paired with a set of five random actions (normally distributed with mean 48 (the higher narrower peak) and standard deviation 10) for the other player, and the **maximum** of the joint rewards is plotted. The process is repeated three times, with one curve for each try.

of agent A from one situation and agent B from the other results in two agents which compete for the resources to neither's benefit.

Teammate learning literature breaks down along different lines than team learning literature. Since each agent is free to learn separately, heterogeneity versus homogeneity has been considered an emergent aspect rather than a design decision in teammate learning (for example, (Balch, 1997, 1999) has argued that the more local a reinforcement signal, the more heterogeneous the final team). Further, relatively little teammate learning work has been done in the area of communication.

Thus we have broken the teammate learning literature down into the following four groups. First: papers which analyze the *optimality of performance* when using teammate learning. The interest stems from the desire to have individual agents learn how to perform as well as possible as a team despite being rewarded individually. Second: papers dealing with the impact of *locality of reward* on the learned behaviors. The locality of reward, directly related to the problem of credit assignment among agents, seems to affect the heterogeneity of the learned behaviors and their performance in different types of domains. Third: papers examining interesting issues in *competition versus cooperation.* Fourth: papers which deal with *modeling other agents* in other to improve the interaction with them.

5.1 Optimal Team Behavior

Many studies in optimal team behavior (or the lack thereof) in teammate learning tend to examine the problem from a game-theoretic perspective.

Repetitive games In a repetitive game, two or more agents interact repeatedly, and after each interaction each agent may receive some reward. When all agents receive the same reward each time (this may be thought of as a "team reward"), the domain is called "fully cooperative" (Boutilier, 1996). Through repeated interactions, each agent adjusts its strategy in order to maximize the received payoff. Boutilier (1996) surveys a number of issues associated with repetitive games, focussing on pathological but generally unrealistic game situations, and

encourages further theoretical analysis. Claus and Boutilier (1998) proposes two benchmark games (*climb* and *penalty*) and shows that convergence to global optima is not always achieved in these games even if agents use reinforcement learning based on the joint-action information (rather than each agent's individual action). This result is disturbing, given the fact that agents had *complete information* about the team and the environment, and provides a strong incentive for developing better multiagent learning algorithms with guarantees on convergence to optimal results.

Lauer and Riedmiller (2000) suggest updating an agent's policy (Q-values) by estimating the likely best cooperation possible for the agent's action, and proves that this will converge to the optimum in deterministic environments. Kapetanakis and Kudenko (2002b,a) point out possible flaws in Lauer's approach when dealing with stochastic domains, and present a modified exploration strategy that improves cooperation under these new conditions. Panait et al. (2003) point out that standard coevolutionary approaches, when applied to Lauer's fully-competitive problem domains, are sometimes driven by "balance" considerations rather than performance. Similar to Lauer, Panait *et al* show that evaluating an agent in the context with agents chosen as estimates to its best possible collaborators yields significantly improved results over the standard coevolutionary methods. Other investigations of learning algorithms for repetitive games are presented in (Banerjee et al., 2000; Chang and Kaelbling, 2001).

Stochastic games Stochastic games are repetitive games in which the game outcome (and corresponding reward per agent) is a stochastic, rather than deterministic, function of the agent's interactions. Bowling and Veloso (2000) examine a number of game theory and reinforcement learning approaches to stochastic games, and describes the differences in assumptions they make. Hu and Wellman (1998a) introduce a reinforcement learning algorithm for solving general-sum stochastic games: this proof is corrected and revised with addditional constraints in Bowling (2000). Bowling and Veloso (2001) describe two desirable properties for learning agents, namely rationality (the agent should converge when the other agents are fixed to stationary strategies) and convergence (under the specified conditions, all agents will converge to stationary strategies), and present a learning algorithm that exhibits these two properties. Bowling and Veloso (2002a) investigate the existence of equilibria points for agents with limitations. Another investigation of learning in multiagent stochastic games is presented in (Bowling and Veloso, 2002b). Most of the work on stochastic games assumes the uniqueness of the globally optimal cooperation strategies; a notable exception is presented in (Suematsu and Hayashi, 2002)

Other Analysis Coevolution may be cast into an evolutionary game theoretic framework, though not exactly as a repetitive or stochastic game. Wiegand has analyzed the conditions under which coevolutionary systems gravitate towards Nash optima rather than providing globally optimal solutions for the team as a whole; and approaches to assessing the external performance of the learners despite their sensitivity to the other agents involved (Wiegand, 1998; Wiegand

et al., 2001, 2002a,b). (Zhao, 1998) presents a societal model to cooperative coevolutionary algorithms. Gordin et al. (1997) report that a shared memory data structure containing the best-discovered-so-far teams of agents yields superior team performance when using the cooperative coevolution in a room-painting domain.

In a series of papers, Schmidhuber *et al* examine the rates of team performance improvement rather than final team performance. Schmidhuber and Zhao (1996) apply the related "success-story algorithm": agents periodically check their performance and undo behavioral modifications that were not observed to lead to lifelong reward improvements.

5.2 Credit Assignment and the Locality of Reward

One facet of the credit assignment problem is the degree to which reward should be local to individual agents. Consider a gold-mining problem, where reward is given to agents based on how much gold they extract. If an agent was rewarded only for his own gold extraction, agents would revert to greedy behaviors, having no incentive to inform other agents of rich lodes that the whole team could work on. On the other hand, if all agents were rewarded equally for any gold extracted, "lazy" agents would have no incentive to mine gold at all — they would be rewarded regardless.

The "laziness" effect generally means that local reward methods tend to result in faster learning rates (Balch, 1997, 1999). But Balch shows that whether local or global reward mechanisms lead to better performance tends to be problem-domain dependent. Balch (1997, 1999) the use of global (R_{global}) and local (R_{local}) reinforcement policies; R_{global} depends on the team performance, while R_{local} depends only on whether the agent scored a goal or not. The results suggest that local rewards lead to better performance in a foraging task (Balch, 1999), but better results are obtained when using global rewards in a simulated soccer domain (Balch, 1997): teams of agents trained using R_{global} outscore on average 6 goals to 4 a fixed control team, while teams of agents trained with R_{local} lose by an average of 4 points to 6.

The previous two papers also suggest that the locality of rewards affects the heteorgeneity (diversity) of the learned behaviors. In both the foraging and simulated soccer domains, using local rewards results in homogeneous behaviors, while global rewards lead to heterogeneous teams. .

In a related approach, Tangamchit et al. (2002) use local and global rewards in combination with average (Monte-Carlo) or discounted (Q-Learning) reinforcement in a foraging domain. The authors suggest that discounting leads to robots learning actions targeting for immediate payoff, hampering better collaborations. At the other extreme, average reinforcement, combined with global reward leads to much better results.

5.3 Cooperation, Competition and In-Betweens

5.4 Teammate Modeling

A final area of research has been in teammate modelling: learning about other agents in the environment so as to make good guesses of their expected behavior, and to act accordingly. As other agents are likely modeling *you*, modeling *them* in turn brings up the spectre of infinite recursion: "Agent A is doing X because it thinks that agent B thinks that agent A thinks that agent B thinks that ..." This must be rationally sorted out in finite time. Vidal and Durfee (1997) categorize agents based on the complexity they assume for their teammates. A 0-level agent believes that none of its teammates is performing any learning activity and it does not consider their changing behaviors as "adaptive" in its model. A 1-level agent models its teammates as 0-level agents; in other words,

it considers his teammates do not model any other agent (including the 1-level agent). In general, an N-level agent models its teammates as (N-1)-level agents.

Mundhe and Sen (2000a) present an initial investigation on the use of 0-level, 1-level and 2-level modeling agents. The authors report a very good performance for the 0-level learners, suggesting that for some domains teammate modeling may not be necessary. Similar results showing good coordination without modeling other agents are reported in (Sen et al., 1994; Sen and Sekaran, 1998), where a couple of robots successfully learn to cooperatively push a box without either even being aware of the other's presence. However Mukherjee and Sen (2001) present an experiment in which 1-level agents model each others' action probability distribution; this produces a form of mutual trust which yields better performance.

When dealing with larger numbers of teammates, another modeling approach is to presume the entire team is formed of homogeneous agents (). (Haynes and Sen, 1996c,a; Haynes et al., 1996) use this approach complemented with a case-based learning mechanism. Nagayuki et al. (2000) present a similar approach for heterogeneous teams

While their work was applied to competitive games, Wellman and Hu (1998); Hu and Wellman (1996) present a result that is important for cooperative modeling as well, namely that when learning models of other agents in a multiagent learning scenario, the resulting behaviors are highly sensitive to the agents' initial beliefs. Depending on these initial beliefs, the final performance may be no better than when *no* teammate modeling is performed — agent modeling may prevent agents from converging to optimal behaviors. A similar conclusion is reported by Hu and Wellman (1998b): the authors suggest the best policy for creating learning agents is to *minimize* the assumptions about the other agents' policies.

Suryadi and Gmytrasiewicz (1999) present an agent modeling approach using *influence diagrams* (similar to belief networks). The approach consists of learning the beliefs, capabilities and preferences of the teammates in order to improve cooperation. As the correct model cannot be usually computed, the system stores a set of such models together with their probability of being correct. Then the set of models is adjusted according to observations about the behaviors of the other agents.

We conclude this section with work in agent modeling under communication. Ohko et al. (1997) use a communication protocol for agents to subcontract subtasks to other agents. Ohko *et al* investigate the use of case-based reasoning to reduce the communication effort for task announcements by enabling agents to acquire and refine knowledge about other agents' task solving abilities. With the embedded learning algorithm, communication is reduced from broadcasting to everyone to communicating exact messages to only those agents that have high probabilities to win the bids for the tasks. A related approach is presented in (Bui et al., 1998, 1999): here, Bayesian learning is used to incrementally update models of other agents to reduce communication load by anticipating their future actions based on their previous ones.

6 Learning and Problem Decomposition

The state space of a large, joint multi-agent task can be overwhelming. An alternative is to reduce complexity by heuristically decomposing the overall task into simpler subtasks that agents can more easily approach. Humans are not perfect, and our application of domain-specific knowledge to perform such a heuristic decomposition could inadvertantly bias the learning procedure so as to remove the optimal areas of the solution space. However, for large and complex problems, the savings in computational complexity are often worth it.

Squads One approach to decomposition is to group agents into squads, with each squad following the same behavior, rather than all the agents learning separate behaviors.[6] These squads are then placed in specific situations, or endowed with different abilities, to encourage the development of separate behaviors believed necessary to solve the joint task. For example, a team of eleven soccer-playing robots might be grouped into squads of forwards, midfielders, attackers, and a goalie (Luke, 1998; Luke et al., 1997). The number of such squads can also be learned, giving the team some freedom in problem decomposition (Hara and Nagao, 1999).

Layered Learning In layered learning, each individual behavior is decomposed into a hierarchy of behaviors, each using the ones beneath it. Some of the behaviors may be hardcoded, while others are modified by learning processes. The method was successfully used to learn behaviors for robotic soccer (Stone, 1998, 1997). Further applications of the technique are reported by Gustafson (2000); Gustafson and Hsu (2001); Hsu and Gustafson (2002). The authors present an interesting application of genetic programming to the Keep-Away Soccer domain, where three learned offensive players face off against a single hard-coded defender who can move at twice their speed. The genotype of an individual encodes a single homogeneous behavior that is applied to each offensive agent in the environment. Gustafson and Hsu compare a layered learning approach to traditional genetic programming techniques in learning this behavior. The authors teach the offensive team the behaviors for several basic subtasks, then teach it to solve a more complex joint behavior which relies on them.

Shaping Shaping consists of decomposing the learning task into a series of incrementally difficult tasks (Singh, 1992). The first task in the series is very simple and the agents can relatively easily and quickly learn it. As soon as that happens, the problem changes to solving the second task in the series, then the third, and so on. The process continues until the agents have learned to perform the most difficult task (and the only relevant one). This differs from layered learning in

[6] Indeed one might argue that a homogeneous multiagent behavior approach is the extreme end of the squad method: by grouping all the agents into a single "squad", the experimenter is essentially heuristically betting that agents need only learn a single unified behavior to solve the problem.

that the previous tasks are not explicitly used as "modules" on which later tasks rely. Instead the problem has been decomposed under the assumption that the system will on its own learn how to modify a solution to an easier task into a solution for a harder task. Balch (1999) introduces a shaped reinforcement reward function (R_{shaped}) (also used in (Mataric, 1997)) which depends on the number of partial steps fulfilled for accomplishing the task. The author shows that using R_{shaped} leads to similar results to using a local reward function R_{local}, but in a significantly shorter time. The results are also better than those obtained when using a global reward function R_{global}.

Team Behavior Decomposition Guestrin et al. (2002) present an interesting approach to applying reinforcement learning to multiagent domains. The authors note that in many domains the actions of some agents may be independent. Taking advantage of this, they suggest creating partially decomposing the joint Q-values of agents based on a *coordination graph* that heuristically spells out which agents must interact in order to solve the problem. This partial decomposition permits a heuristic middle-ground between learning a full joint utility table and learning separate independent tables. The authors propose a series of coordinated reinforcement algorithms by which agents coordinate both their action selection and policy updates.

7 Learning and Communication

To solve some problems, it can be critical for agents to communicate; for others, communication may still increase agent performance. We define *communication* very broadly: altering the state of the environment such that other agents can perceive the modification and decode information from it. Among other reasons, agents communicate in order to coordinate more effectively, to distribute more accurate models of the environment, and to learn subtask solutions from one another.

But are communicating agents really *multi-agent?* Stone and Veloso (2000) have noted that large, unrestricted communication also reduces a multiagent system to something isomorphic to a single-agent system. They do this by noting that this permits all the agents to send complete external state information to a "central agent", and to execute its command in lock-step, in essence acting as effectors for a single agent. A central agent is not even necessary: as long as agents can receive all the information they need about the current states of all the other agents, they can make independent decisions knowing exactly what the other agents will do, in essence enabling a "central controller" on-board each individual agent, picking the proper sub-action for the full joint action. Thus we feel that a true multi-agent problem necessitates restrictions on communication. At any rate, while full, unrestricted communication can orthogonalize the learning problem into a basic single-agent problem, such an approach requires very fast communication of large amounts of information. Real-time applications instead place considerable restrictions on communication, in terms of both bandwidth and speed.

Explicit communication can also significantly increase the learning method's search space, both by increasing the size of the external state available to the agent (it now knows state information communicated from other agents), and by increasing the agent's available choices (perhaps by adding a "communicate to agent i" action). As noted in (Durfee et al., 1987), this increase in search space can hamper learning an optimal behavior more than communication itself can help. Thus even when communication is required for optimal performance, for many applications the learning method must disregard communication, or hard-code it, in order to simplify the learning process. For example, when learning in a predator-prey pursuit domain, Luke and Spector (1996) assume that predators can sense each others position no matter what distance separates them, and Berenji and Vengerov (2000b) use a blackboard communication scheme to allow agents to know of other agents' locations.

7.1 Communicating to Improve Team Performance

Learning agents may communicate in order to improve team performance, commonly through discovering how to signal one another during the task. For example, Yanco and Stein (1993) experiment with two mobile robots application in a cooperative movement task, and endowed with a fixed but undefined communication vocabulary. The two robots have an initial fixed and uninterpreted communication vocabulary. The robots learn to associate different meanings to words in different trials. Additionally, when circumstances change, the robots learn to adjust their communication language (or their behaviors) to the new situations.

A similar approach is reported in (Jim and Giles, 2000), where a genetic algorithm is used for language learning in a predator-prey domain. The authors use the blackboard communication scheme and show that increasing the language size improves the performance. Additionally, they show that evolved communicating predators perform better than all previous reported results and present a rule for determining a pessimistic estimate on the minimum language size that should be used for multiagent problems. A particularly active researcher in the evolution of communication is Luc Steels at the Free University of Brussels. Experiments he reports in (Steels, 1996a) show the emergence of a spontaneous coherent lexicon that may adapt to cope with new meanings during the lifetime of the agents. Steels and Kaplan (1999) continue this investigation on evolving the communication and shows that agents are able to create general words by collective agreement on their meanings and coverage. Similar approaches to evolving communication languages are presented in (Steels, 1995; Saunders and Pollack, 1996; Steels, 1996b,c, 1997, 2000; Cangelosi, 2001).

7.2 Communicating to Speed Learning

Some research, particularly in reinforcement learning, has presumed that the agents have access to a joint utility table or to a joint policy table to which each may contribute in turn, even though the agents are separate learners. We

argue that this is an implicit hard-coded communication procedure: the learners are teaching each other learned information. For example, Berenji and Vengerov (2000a) argue that multiple Q-learners sharing a joint Q table can learn significantly faster than independent learners.

Teaching can also be done implicitly, and can be mixed with signaling. Tan (1993) uses a predator-prey pursuit domain and shows that teams with learning agents that share knowledge about the task significantly outperform independent learners. The sharing of knowledge is done in three ways: sharing immediate sensor information (state), sharing episodes (sequences of $\langle state, action, reward \rangle$ experienced by an agent), or sharing policy information (in the form of $\langle state, action, utility \rangle$).All learning benefits are obtained at the expense of increased communication usage.

7.3 Pheromones: Communicating via Slow, Locally-sensed State Changes

Pheromones permit agents to leave markers in the local environment which last a long time, as opposed to the rapid- and global- information methods discussed above. This is analagous to Hansel and Gretel's leaving bread crumbs in order to find their way back home while wandering in the forest. Most work in this area is inspired by social insects such as ants and termites, which communicate by laying pheromones on the ground: pheromones are chemical compounds whose presence and concentration can be sensed by the insects (Bonabeau et al., 1999). For example, while foraging for food, an ant drops pheromones to mark the trail connecting the nest to food source (Hölldobler and Wilson, 1990). The ant uses pheromone trails both to find its way back to the food source, and also to recruit other ants to forage from food source. In some situations, pheromones may diffuse or evaporate.

In some sense, pheromone deposits can be seen as a giant blackboard or state-utility table shared by all the agents; it is different from this in that pheromones can only be detected locally. Pheromones are not a random-access memory. Nonetheless, the similarities between pheromone information and other joint memory mechanisms is interesting. Pheromones can be (more or less) static until updated by other agents. Agents can often disambiguate among similar states by storing different amounts or different types of pheromones. Agents can also propose generalizations over classes of similar states by depositing similar amounts of pheromones at those locations.

Several pheromone-based learning algorithms have been proposed for foraging problem domains. A series of reinforcement learning algorithms have adopted a fixed pheromone laying procedure, and use current pheromone amounts as additional sensor information while exploring the space or while updating the state-action utility estimates (Leerink et al., 1995; Monekosso and Remagnino, 2001; Monekosso et al., 2001, 2002; Monekosso and Remagnino, 2002). Evolutionary computation techniques have also been applied to learn exploration/exploitation strategies using pheromones deposited by hardcoded mechanisms. For example, (Sauter et al., 2001, 2002) show how EC can be used to tune an agent policy in

an application involving multiple digital pheromones. A similar idea applied to network routing is presented in (White et al., 1998).

An interesting research question is whether agents can learn not only to use pheromone information but to deposit the pheromones in a rational manner. This question was first examined in AntFarm, a system that combines communication via pheromones and evolutionary computation (Collins and Jefferson, 1992, 1991). AntFarm uses multiple colonies of homogeneous ants, each colony in a separate 16x16 grid environment. The ants use a single pheromone to mark trails toward food sources, but use a compass to orient themselves on the shortest path back to the nest. The system uses a neural network representation and plans to evolve only the foraging strategies, while the ones for returning home are hardcoded. In the future work section of the paper, Collins and Jefferson admit not having observed the evolution of cooperative behaviors (Collins and Jefferson, 1992). The authors suggest the problem comes from the incapability of the system to discover how to create nice uphill gradients of pheromones for ants to use during the foraging process.

Panait and Luke (2004a) present an algorithm that exhibits good performance at various foraging tasks. This algorithm uses multiple pheromones, and in doing so it eliminates the explicit requirement for ants to precisely know the direction towards the nest from any point. Rather, ants use pheromone information to both guide themselves back to the food source and to the nest. The new algorithm exhibits good performance in the presence of obstacles. Additionally, the authors mention an interesting hill-climbing effect that leads to straight (and locally optimal) paths. In an extension of their work, Panait and Luke (2004b) successfully apply evolutionary computation to learn a pheromone-based foraging task. In a series of three experiments, the authors show that a colony of agents can learn to both deposit pheromones and to use that information in successful foraging behaviors. A comparison of the results on increasingly complicated environments shows that behaviors learned for more complex domains exhibit good performance in simpler problems as well.

7.4 Communication via Embodiment

In many domains, agents are acting as "brains" that control "bodies" performing specific tasks. This nearly always is the case in robotics applications, where the robots (in simulation or in reality) act as the bodies, while the agents are software or humans that control them. In such domains, agents can sense (via the sensors located on the robots) other nearby robots. This makes it possible for agents to communicate with each other by just changing or setting the position of their respective robots. For example, by positioning the robot in the middle of a doorway, an agent can signal to other robots that they should avoid the room.

Quinn (2001b) argues that this is a form of communication by citing Wilson's statement that communication "is neither the signal by itself, nor the response, it is instead the relationship between the two" (Wilson, 1975). This definition suggests that a shared dedicated channel is not necessary for communication.

Quinn (2001b) investigates evolution of strategies in a two-agent collaborative-movement domain and reports that robots were able to coordinate by communicating their adopted roles of leader or follower via sequences of moves. For example, after initial phases of alignment, the robots use rotation and oscillatory back-and-forth movements to decide who leads the way and who follows. Quinn terms this "communicative behavior", because it is a sequence of actions meant to convey information from one agent to the other.

8 Two Challenges: Scalability and Adaptive Dynamics

Multiagent learning is a new field and as such its open research issues are still very much in flux. However we believe that two specific areas have proven themselves important challenges to overcome in order to make multiagent learning more broadly successful as a technique. Both of these challenges arise from the *multi* in multiagent learning, and may eventually require new learning methods special to multiple agents, as opposed to the more conventional single-agent learning methods (case-based learning, reinforcement learning, traditional evolutionary computation) now common in the field.

8.1 Scalability

Scalability is a problem for many learning techniques, but especially so in multiagent learning. The dimensionality of the search space grows rapidly with the complexity of possible agent behaviors, the number of agents involved, and the size of the network of interactions between them. This search space grows so rapidly that it seems clear that one *cannot* learn the entire joint behavior of a large, heterogeneous, strongly intercommunicating multiagent system. Effective learning in an area this complex requires some degree of sacrifice: either by isolating the learned behaviors among individual agents, by reducing the heterogeneity of the agents, or by reducing the complexity of the agent's capabilities. Techniques such as learning hybrid teams, or partially restricting the locality of reinforcement, provide promising solutions in this direction, but it is not well understood under which constraints and for which problem domains these restricted methods will work best.

8.2 Adaptive Dynamics

Multiagent systems are typically dynamic environments, with multiple learning agents vying for resources and tasks. This dynamicism presents a unique challenge not typically found in single-agent learning: as the agents learn, their adapting to one another changes the world scenario. How do agents learn in an environment where the goalposts are constantly being moved? This dynamicism also presents the interesting problem of quality assessment. In a decentralized domain, such quality assessment is relative to or in the context of other agents in

the environment. Thus in many cases there is *no absolute quality measure* that can be assigned to an agent.

Co-adapting multiagent systems may be thought of in game-theoretic terms, suggesting a convergence to stable equillibria; as opposed to actual optima. Researchers, especially coevolutionary theorists, have used Nash equilibrium points to investigate the learning algorithms for multiagent systems, and have discovered much of the art of cooperative multiagent learning is figuring out how to warp the space so that equilibria are likely to be near-optimal. For example, cooperative coevolution has revealed that the multiagent learning process is sometimes guided by "balance" considerations, rather than performance improvements. The adaptation of single-agent learning algorithms to the multiagent domain neglects an important fact: how well an individual performs in a *typical* team is not well-related to the individual's performance in a *good* team, much less an optimal one. Panait et al. (2003) showed that when an agent is evaluated in the context of its ideal collaborators, the dynamics of a coevolutionary appracoh reduce to a simple evolutionary algorithm for learning multiagent teams. That is, the closer one can get to assessing individuals with their optimal collaborators, the more likely one was to optimize the system in an evolutionary computation sense.

9 Problem Domains and Applications

Despite the relative young age of the field, the multiagent systems area contains a very large number of problem domains. The list in this survey is by far complete, but it contains many of the benchmark problems. The problem domains are divided in three classes: cooperative robotics, game-theoretic environments, and applications to complex real-world problems.

9.1 Embodied Agents

The cost of robots have decreased significantly, making it feasible to purchase and use several (tens, hundreds, or even thousands of) robots for a variety of tasks. This drop in cost has spurred research in multiagent cooperative robotics. Additionally, computer hardware is cheap enough that what cannot be performed with real robots can now be done in simulation; though the robotics community still strongly encourages validation of results on real robots.

This sections lists several problem domains from the cooperative robotics or simulation community. Most of them are well-known benchmark problems for multiagent learning techniques.

Predator-Prey Pursuit This is one of the most common environments in multiagent learning research, probably because it is easy to implement. Pursuit games consist of a number of agents (predators) cooperatively chasing a prey. Individual predator agents are usually not faster than the prey, and often agents can

sense the prey only if it is close by. Therefore, the agents need to actively cooperate in order to successfully capture the prey. Our earliest known work using this domain is (Benda et al., 1986), but many variations have been studied since: discrete and continuous environments; variable numbers of predators, available sensing and communication; time constraints, etc).

Foraging The domain consists of a large map with agents (robots) and items to forage (pucks or cans). The task is to carry the items to specially designated areas. Variations may include multiple item locations, item types, or drop locations. The efficiency of an approach can be defined by how quickly it completes the foraging task (Mataric, 1994), or by the number of items collected in a fixed amount of time. Ostergaard et al. (2001) provide an extended taxonomy accompanied by useful examples of previous work. Teams of agents using either local or global reward information have been shown to have similar performances (Balch, 1999), suggesting that locally optimal (greedy) agent behaviors can be composed to form good performing teams.

Box Pushing This domain involves a two-dimensional bounded area containing obstacles and a number of boxes. Agents in this environment need to arrange the boxes to specified final positions by moving near them and pushing them. Sometimes a robot is capable of pushing one box by itself (see for example the single-agent experiments reported in (Mahadevan and Connell, 1991)). A more complicated version requires two or more agents to cooperate in order to move a single box in simulation (Zhang and Cho, 1998), or on real robots (Mataric et al., 1995).

Soccer The game of soccer, both in simulation and with real robots, is one of the most widely used domains for recent research in multiagent systems (Kitano et al., 1997). The domain consists of a soccer field with two goals, a ball, and two teams with a number of agents (from 5 to 11, depending on the sizes of the robots and of the field). The performance of a team is usually assessed based on the difference in number of goals scored opposing teams. Other performance metrics have included length of time in control of the ball, successful interceptions, and average location of the ball. . The strong interest in this domain has led to several annual "world cup" robot soccer championships. The most well-known such competition, RoboCup (*www.robocup.org*), has different leagues divided by continent and by type of robot or simulation environment. RoboCup's goal is to "develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team".

Keep-Away Soccer Gustafson and Hsu (2001) use a simpler version of the *Soccer* domain which contains only one offensive and three defensive players and a ball. The defensive player is twice as fast than the offensive ones, and the ball, when passed, can move at twice the speed of the defensive player. The objective is to keep the ball away from the defensive player by moving and passing; there is a penalty each time the defensive player is within one grid unit away from the ball. A similar domain is presented in (Stone and Sutton, 2002).

Cooperative Navigation The task, as described in (Balch, 1998), is to have a team of agents move across a field in minimal time without colliding with obstacles or other robots. Each agent selects from a number of predefined strategies, and the performance is assessed based on the maximum time necessary for the agents to accomplish the task. The questions investigated in this thesis include benefits from formation participation and impact of diversity on performance.

Cooperative Target Observation Introduced in (Parker, 2000b), this domain involves a two dimensional bounded area in which a number of robots have to keep several moving targets under observation. Performance is based on the total number of targets within the "observable" distance to any team-member robot during the time period. Parker investigates an initial approach to learning behaviors for this domain and reports improvements over naive, random approaches, but also notes the superiority of the hand generated solutions. Additional research investigations using this domain include (Fernandez and Parker, 2001; Parker et al., 2001; Parker, 2002).

Herding (Schultz et al., 1996) introduces a new domain where a robot must herd another robot into a designated area. The second robot (the sheep) tries to avoid obstacles, but can either move randomly or try to avoid the herding area. The herder moves close to the sheep robot to encourage the sheep to move in the desired directions. Later investigations (including (Potter et al., 2001)) try a multi-shepherd version of the domain, where many faster and "stubborn" sheep require coordination from the several herding agents. Additionally, predators ("foxes") exist in the environment and try to kill the sheep while avoiding the shepherds, thus complicating the shepherds' task.

9.2 Game-Theoretic Environments

As discussed in Section 8.2, many multiagent systems may be cast in game-theoretic terms; essentially as strategy games consisting of matrices of payoffs for each agent based on their joint actions. In addition to game-theoretic analysis of multiagent systems, some common problem domains are also taken from game theory.

Iterated Prisoners' Dilemma In the classic Prisoner's Dilemma domain, two prisoners are questioned about the crime they jointly committed. Each has the opportunity to either cooperate with the other (not say anything) or to defect (squeal on him) without knowing about the other agent's action. The reward or punishment for cooperation or defection is described numerically, with higher numbers being better. If both cooperate, they each receive a reward of 3. If one defects and the other cooperates, the defector receives a reward of 5, while the cooperator receives 0. If both defect, they each receive a reward of 1. In the iterated version of the game, the scenario repeats a number of times, enabling the agents to each learn from the other's behavior and adapt their actions appropriately. The Iterated Prisoner's Dilemma is considered a *cooperative* game because the

better pairs of agents tend to cooperate with one another (Axelrod, 1984, 1987). A three-person coordination game inspired by the Iterated Prisoner's Dilemma is presented in (Akiyama and Kaneko, 1995).

Social Dilemmas These problems concern individual decisions of several agents, all of which receive a joint reward (Glance and Huberman, 1994). The *Tragedy of the Commons*, *Braess Paradox* and *Santa Fe Bar* are examples of social dilemma games. In the Tragedy of the Commons, a number of agents share a resource of limited capacity. When the joint usage of the resource exceeds the capacity, the service deteriorates, and so do the rewards received by the agents. In the Braess Paradox problem, agents share two resources. The dilemma arises when agents must decide to start accessing the less utilized of resources: if all agents decide to do so, it will become overwhelmed and rewards will drop. Further details on the two problems, accompanied by a coevolutionary approach to learning solutions to them, can be found in (Mundhe and Sen, 2000b). In the Santa Fe Bar problem, a large number of agents individually must decide whether to go to a bar in Santa Fe. If too many or too few agents opt to go, their satisfaction is lower than when a reasonable number of them decide to go (Arthur, 1994).

9.3 Real-World Applications

This section describes a set of problems inspired from real-world domains. The problems described in this section have been used previously in MAS investigations. Many of the described problem domains are logistics, planning, and constraint-satisfaction problems requiring real-time, distributed decision making. Because the applications are often very complex and highly distributed, learning techniques have rarely been applied to them, and so they are presented here as example challenge problems for multi-agent learning.

Distributed Vehicle Monitoring The task in this domain is to maximize the throughput of cars through a grid of intersections. Each intersection is equipped with a traffic light controlled by an agent. The agents need to coordinate to deal with fluctuations in traffic (Lesser et al., 1987).

Air Traffic Control For security purposes, the air space is divided into three-dimensional regions used by air traffic controllers to guide the airplanes to their final destination. Each such region has an upper bound (called the *sector capacity*) on the number of airplanes it can hold at any time. The task is to guide the planes from sector to sector along minimal length routes, while ensuring that constraints are met; the solution needs to be fast to handle real-time data. A multiagent approach for this domain is reported in (Steeb et al., 1988).

Network Management This domain consists of a large, distributed network of interacting entities. Agents are deployed in network infrastructures in order to distributively and cooperatively control and manage the network, handle failures, and balance its load (Weihmayer and Velthuijsen, 1994).

Electricity Distribution Management Here the problem is to maintain an optimal power grid configuration that keeps all customers supplied and minimizes losses; while at the same time dealing with possible damage to the network, variable demand from customers, scheduled maintenance operations, and equipment failures and upgrades (Varga et al., 1994).

Distributed Medical Care This domain applies AI to assist clinical staff in making diagnoses, decide on therapy and tests, determine prescriptions, and other matters related to medical care ((Huang et al., 1995)). The problem is particularly suited for multiagent systems because of the decentralization of data and resources, high costs for obtaining comprehensive information, and stochasticity and dynamicity of data.

Factory Production Sequencing This classic planning and scheduling problem involves managing the process of producing complex items through a series of steps, where there are different constraints and costs associated with each step. The task consists of building a plan (a *production sequence*) that specifies the order of operations for different items such that the production costs are minimized while satisfying customer orders (Wooldridge et al., 1996).

Hierarchical Multi-Agent Systems Problems Some multiagent domains are of particular interest because of the different levels at which problems can be formulated. For example, in the "Transportation" problem, several trucking companies transport goods between locations. Depending on problem formulation, agents can represent whole companies, sets of trucks from the same or from different companies, or even individual trucks. The task is to complete requests from customers under specific constraints (maximum time required to finish the job, minimal cost of delivery, etc.). A multiagent approach to this domain is reported in (Fischer et al., 1993). The "Loading Dock" is a similar problem, where several forklifts load and unload trucks according to task requirements; either individual forklifts or groups of forklifts may be modeled as an agent (Muller and Pischel, 1994). A related "Factory Floor' problem is investigated in (Peceny et al., 1996), where products are manufactured from raw material by several machines under time and cost minimization constraints, and agents can represent individual machines or groups of machines.

Models of Social Interaction Many natural and social systems have very large numbers of interacting agents. Accordingly, such interactions need also be present in simulations of these natural phenomena. Examples of interesting work in this area include Craig Reynolds' work on collective behaviors such as flocking Reynolds (1987); learning behaviors for complex artificial creatures in virtual worlds, such as those in the game of Creatures (Grand et al., 1997); and the modeling of the formation of early countries Cederman (1997).

Other multiagent systems domains investigated include particle accelerator control (Jennings et al., 1993), intelligent document retrieval (Mukhopadjyay et al.,

1986), spacecraft control (Schwuttke and Quan., 1993), concurrent engineering (Cutkosky et al., 1997), job-shop scheduling (Morley and Schelberg, 1993), steel coil processing (Mori et al., 1988). Further applications of Distributed AI in industry are reported in (Van Dyke Parunak, 1996).

10 Resources

Machine Learning in Multiagent Systems is a relatively new research topic, with a large increase in the number of research papers published in the last few years. General material on the topic can be found in (Weiß, 1999, 1997, 1995; Imam, 1996; Weiß and Sen, 1996; Sen, 1996; Huhns and Weiß, 1998; Weiß, 1998; Sen, 1998). Other surveys of existing work in multiagent learning and related domains include (Stone and Veloso, 2000; Weiß and Dillenbourg, 1999; Van Dyke Parunak, 1996; Brazdil et al., 1991; Cao et al., 1997; Decker et al., 1989, 1999; Durfee, 1992; Durfee et al., 1989; Lesser, 1999; Parker, 2000a; Sen, 1997; Luck et al., 1998; Weiß, 1994). There are also several PhD theses investigating different aspects of the multiagent learning field: (Mataric, 1994; Balch, 1998; Potter, 1997; Gmytrasiewicz, 1992; Carmel, 1997; Stone, 1998; Crites, 1996; Grecu, 1997; Nagendra-Prasad, 1997; Dowell, 1995) Good pointers for news, conferences and journals, courses, laboratories, people, companies and laboratories, may be found on web sites such as http://www.multiagent.com and http://agents.umbc.edu.

11 Conclusions

This paper surveyed the multiagent learning area, presenting a novel categorization of previous work, accompanied by discussions on key aspects of interest. We argued that learning in multiagent systems can be done at a higher "team" level, where a single learner improves the behavior of the entire team, but also at a finer "teammate" level, where individual agents conduct their own learning processes in order to better fit the team.

In particular, Team Learning has received a significant share of interest. It has been previously neglected as a special category of learning in multiagent systems. We presented approaches to learn behaviors for homogeneous, heterogeneous and hybrid homogeneous/heterogeneous teams, accompanied by several comparative investigations of these approaches.

The second part of the paper surveyed Teammate Learning. We surveyed papers investigating the optimality of the learning approaches, the impact of locality of reward information, cooperation or competition relations among agents, and modeling other agents in the domain.

Later sections dealt with other issues of interest related to problem decomposition and communication. In terms of communication, we divide the work according to the mechanism used for communication. The first class includes techniques using regular communication, such as via fiber optic, cables, or wireless. A second set of methods use slowly decaying information such as pheromones; this mechanism allows for longer duration of the information, and it can be used

as a shared memory capacity where agents read and write information associated with different states of the environment. A last class of papers deals with communication via embodiment in domains such as robotics where agents act as brains for actual bodies.

12 Acknowledgements

Significant help in understanding the intricate complexities of multiagent systems came from discussions with Paul Wiegand, Gabriel Balan, Jeff Bassett, Valentin Prudius, Jayshree Sarma, Marcel Barbulescu, Adrian Grajdeanu and Elena Popovici. Valentin Prudius and Jayshree Sarma also helped correcting initial versions of the manuscript.

Bibliography

Akiyama, E. and Kaneko, K. (1995). Evolution of cooperation, differentiation, complexity, and diversity in an iterated three-person game. *Artificial Life*, 2(3):293–304.

Andre, D., Bennett III, F., and Koza, J. (1996). Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. In *Genetic Programming 1996: Proceedings of the First Annual Conference*. MIT Press.

Andre, D. and Teller, A. (1999). Evolving team Darwin United. In Asada, M. and Kitano, H., editors, *RoboCup-98: Robot Soccer World Cup II*. Springer Verlag.

Angeline, P. and Pollack, J. (1993). Competitive environments evolve better solutions for complex tasks. In Forrest, S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA)*, pages 264–270, San Mateo, CA. Morgan Kaufmann.

Arthur, W. (1994). Inductive reasoning and bounded rationality. *Complexity in Economic Theory*, 84(2):406–411.

Axelrod, R. (1984). *The Evolution of Cooperation*. Basic Books.

Axelrod, R. (1987). The evolution of strategies in the iterated prisoner's dilemma. In Davis, L., editor, *Genetic Algorithms and Simulated Annealing*. Morgan Kaufmann.

Balch, T. (1997). Learning roles: Behavioral diversity in robot teams. Technical Report GIT-CC-97-12, Georgia Institute of Technology.

Balch, T. (1998). *Behavioral Diversity in Learning Robot Teams*. PhD thesis, College of Computing, Georgia Institute of Technology.

Balch, T. (1999). Reward and diversity in multirobot foraging. In *IJCAI-99 Workshop on Agents Learning About, From and With other Agents*.

Banerjee, B., Mukherjee, R., and Sen, S. (2000). Learning mutual trust. In *Working Notes of AGENTS-00 Workshop on Deception, Fraud and Trust in Agent Societies*, pages 9–14.

Bassett, J. and De Jong, K. (2000). Evolving behaviors for cooperating agents. In Ras, Z., editor, *Proceedings from the Twelfth International Symposium on Methodologies for Intelligent Systems*, pages 157–165, Charlotte, NC. Springer-Verlag.

Bassett, J. K. (2002). A study of generalization techniques in evolutionary rule learning. Master's thesis, George Mason University, Fairfax VA, USA.

Benda, M., Jagannathan, V., and Dodhiawala, R. (1986). On optimal cooperation of knowledge sources — an empirical investigation. Technical Report BCS-G2010-28, Boeing Advanced Technology Center, Boeing Computer Services.

Berenji, H. and Vengerov, D. (2000a). Advantages of cooperation between reinforcement learning agents in difficult stochastic problems. In *Proceedings of 9th IEEE International Conference on Fuzzy Systems*.

Berenji, H. and Vengerov, D. (2000b). Learning, cooperation, and coordination in multi-agent systems. Technical Report IIS-00-10, Intelligent Inference Systems Corp., 333 W. Maude Avennue, Suite 107, Sunnyvale, CA 94085-4367.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. SFI Studies in the Sciences of Complexity. Oxford University Press.

Bongard, J. C. (2000). The legion system: A novel approach to evolving heterogeneity for collective problem solving. In Poli, R., Banzhaf, W., Langdon, W. B., Miller, J. F., Nordin, P., and Fogarty, T. C., editors, *Genetic Programming: Proceedings of EuroGP-2000*, volume 1802, pages 16–28, Edinburgh. Springer-Verlag.

Boutilier, C. (1996). Planning, learning and coordination in multiagent decision processes. In *Proceedings of the Sixth Conference on Theoretical Aspects of Rationality and Knowledge (TARK96)*, pages 195–210.

Bowling, M. (2000). Convergence problems of general-sum multiagent reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 89–94. Morgan Kaufmann, San Francisco, CA.

Bowling, M. and Veloso, M. (2000). An analysis of stochastic game theory for multiagent reinforcement learning. Technical Report CMU-CS-00-165, Computer Science Department, Carnegie Mellon University.

Bowling, M. and Veloso, M. (2001). Rational and convergent learning in stochastic games. In *Proceedings of Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 1021–1026.

Bowling, M. and Veloso, M. (2002a). Existence of multiagent equilibria with limited agents. Technical Report CMU-CS-02-104, Computer Science Department, Carnegie Mellon University.

Bowling, M. and Veloso, M. (2002b). Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250.

Brazdil, P., Gams, M., Sian, S., Torgo, L., and van de Velde, W. (1991). Learning in distributed systems and multi-agent environments. In Kodratoff, Y., editor, *Lecture Notes in Artificial Intelligence, Volume 482*, pages 412–423. Springer-Verlag.

Bui, H., Venkatesh, S., and Kieronska, D. (1998). A framework for coordination and learning among team of agents. In Wobcke, W., Pagnucco, M., and Zhang, C., editors, *Agents and Multi-Agent Systems: Formalisms, Methodologies and Applications, Lecture Notes in Artificial Intelligence, Volume 1441*, pages 164–178. Springer-Verlag.

Bui, H., Venkatesh, S., and Kieronska, D. (1999). Learning other agents' preferences in multi-agent negotiation using the Bayesian classifier. *International Journal of Cooperative Information Systems*, 8(4):275–294.

Bull, L. and Holland, O. (1997). Evolutionary computing in multiagent environments: Eusociality. In *Proceedings of Seventh Annual Conference on Genetic Algorithms*.

Burke, E., Gustafson, S., and Kendall, G. (2002). A puzzle to challenge genetic programming. In Foster, J. A., Lutton, E., Miller, J., Ryan, C., and Tettamanzi, A. G. B., editors, *Genetic Programming, Proceedings of the Fifth*

European Conference, EuroGP 2002, volume 2278 of *LNCS*, pages 238–247, Kinsale, Ireland. Springer-Verlag.

Cangelosi, A. (2001). Evolution of communication and language using signals, symbols, and words. *IEEE Transactions on Evolutionary Computation*, 5(2):93–101.

Cao, Y. U., Fukunaga, A. S., and Kahng, A. B. (1997). Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4(1):7–23.

Carmel, D. (1997). *Model-based Learning of Interaction Strategies in Multi-agent systems*. PhD thesis, Technion — Israel Institute of Technology.

Cederman, L.-E. (1997). *Emergent Actors in World Politics: How States and Nations Develop and Dissolve*. Princeton University Press.

Chang, Y.-H. and Kaelbling, L. (2001). Playing is believing: The role of beliefs in multi-agent learning. In *Proceedings of Neural Information Processing Systems: Natural and Synthetic (NIPS-01)*.

Claus, C. and Boutilier, C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of National Conference on Artificial IntelligenceAAAI/IAAI*, pages 746–752.

Cliff, D. and Miller, G. F. (1995). Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations. In *Proceedings of the Third European Conference on Artificial Life*, pages 200–218. Springer-Verlag.

Collins, R. and Jefferson, D. (1991). An artificial neural network representation for artificial organisms. In Schwefel, H.-P. and Männer, R., editors, *Parallel Problem Solving from Nature: 1st Workshop (PPSN I)*, pages 259–263, Berlin. Springer-Verlag.

Collins, R. and Jefferson, D. (1992). AntFarm : Towards simulated evolution. In Langton, C., Taylor, C., Farmer, J. D., and Rasmussen, S., editors, *Artificial Life II*, pages 579–601. Addison-Wesley, Redwood City, CA.

Crites, R. H. (1996). *Large-Scale Dynamic Optimization Using Teams of Reinforcement Learning Agents*. PhD thesis, University of Massachusetts Amherst.

Cutkosky, M. R., Englemore, R. S., Fikes, R. E., Genesereth, M. R., Gruber, T. R., Mark, W. S., Tenenbaum, J. M., and Weber, J. C. (1997). PACT: An experiment in integrating concurrent engineering systems. In Huhns, M. N. and Singh, M. P., editors, *Readings in Agents*, pages 46–55. Morgan Kaufmann, San Francisco, CA, USA.

Das, R., Mitchell, M., and Crutchfield, J. (1994). A genetic algorithm discovers particle-based computation in cellular automata. In *Parallel Problem Solving from Nature III, LNCS 866*, pages 344–353. Springer-Verlag.

Davis, J. and Kendall, G. (2002). An investigation, using co-evolution, to evolve an awari player. In *Proceedings of 2002 Congress on Evolutionary Computation (CEC2002)*.

Decker, K., Durfee, E., and Lesser, V. (1989). Evaluating research in cooperative distributed problem solving. In Gasser, L. and Huhns, M., editors, *Distributed Artificial Intelligence Volume II*, pages 487–519. Pitman Publishing and Morgan Kaufmann.

Decker, K., Fisher, M., Luck, M., Tennenholtz, M., and UKMAS'98 Contributors (1999). Continuing research in multi-agent systems. *Knowledge Engineering Review*, 14(3):279–283.

Dowell, M. (1995). *Learning in Multiagent Systems*. PhD thesis, University of South Carolina.

Durfee, E. (1992). What your computer really needs to know, you learned in kindergarten. In *National Conference on Artificial Intelligence*, pages 858–864.

Durfee, E., Lesser, V., and Corkill, D. (1987). Coherent cooperation among communicating problem solvers. *IEEE Transactions on Computers*, C-36(11):1275–1291.

Durfee, E., Lesser, V., and Corkill, D. (1989). Trends in cooperative distributed problem solving. *IEEE Transactions on Knowledge and Data Engineering*, KDE-1(1):63–83.

Fernandez, F. and Parker, L. (2001). Learning in large cooperative multi-robot domains. *International Journal of Robotics and Automation*, 16(4):217–226.

Fischer, K., Kuhn, N., Muller, H. J., Muller, J. P., and Pischel, M. (1993). Sophisticated and distributed: The transportation domain. In *Proceedings of the Fifth European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'93)*.

Fogel, D. (2001). *Blondie24: Playing at the Edge of Artificial Intelligence*. Morgan Kaufmann.

Glance, N. and Huberman, B. (1994). The dynamics of social dilemmas. *Scientific American*, 270(3):76–81.

Gmytrasiewicz, P. (1992). *A Decision-Theoretic Model of Coordination and Communication in Autonomous Systems (Reasoning Systems)*. PhD thesis, University of Michigan.

Gordin, M., Sen, S., and Puppala, N. (1997). Evolving cooperative groups: Preliminary results. In *Working Papers of the AAAI-97 Workshop on Multiagent Learning*, pages 31–35.

Grand, S., Cliff, D., and Malhotra, A. (1997). Creatures : Artificial life autonomous software agents for home entertainment. In *Proceedings of the First International Conference on Autonomous Agents (Agents-97)*, pages 22–29.

Grecu, D. L. (1997). *Using Learning to Improve Multi-Agent Systems for Design*. PhD thesis, Worcester Polytechnic Institute.

Grefenstette, J. (1991). Lamarckian learning in multi-agent environments. In Belew, R. and Booker, L., editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 303–310, San Mateo, CA. Morgan Kaufman.

Guestrin, C., Lagoudakis, M., and Parr, R. (2002). Coordinated reinforcement learning. In *Proceedings of the 2002 AAAI Symposium Series: Collaborative Learning Agents*.

Gustafson, S. M. (2000). Layered learning in genetic programming for a cooperative robot soccer problem. Master's thesis, Kansas State University, Manhattan, KS, USA.

Gustafson, S. M. and Hsu, W. H. (2001). Layered learning in genetic programming for a co-operative robot soccer problem. In Miller, J. F., Tomassini, M., Lanzi, P. L., Ryan, C., Tettamanzi, A. G. B., and Langdon, W. B., editors, *Genetic Programming: Proceedings of EuroGP-2001*, volume 2038, pages 291–301, Lake Como, Italy. Springer-Verlag.

Hara, A. and Nagao, T. (1999). Emergence of cooperative behavior using ADG; Automatically Defined Groups. In *Proceedings of the 1999 Genetic and Evolutionary Computation Conference (GECCO-99)*, pages 1038–1046.

Harvey, I., Husbands, P., Cliff, D., Thompson, A., and Jakobi, N. (1997). Evolutionary robotics : the Sussex approach. *Robotics and Autonomous Systems*, 20(2–4):205–224.

Haynes, T., Lau, K., and Sen, S. (1996). Learning cases to compliment rules for conflict resolution in multiagent systems. In Sen, S., editor, *AAAI Spring Symposium on Adaptation, Coevolution, and Learning in Multiagent Systems*, pages 51–56.

Haynes, T. and Sen, S. (1995). Evolving behavioral strategies in predators and prey. In Weiß, G. and Sen, S., editors, *Adaptation and Learning in Multiagent Systems*, Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin, Germany.

Haynes, T. and Sen, S. (1996a). Adaptation using cases in cooperative groups. In Imam, I., editor, *Working Notes of the AAAI-96 Workshop on Intelligent Adaptive Agents*, Portland, OR.

Haynes, T. and Sen, S. (1996b). Cooperation of the fittest. Technical Report UTULSA-MCS-96-09, The University of Tulsa.

Haynes, T. and Sen, S. (1996c). Learning cases to resolve conflicts and improve group behavior. In Tambe, M. and Gmytrasiewicz, P., editors, *Working Notes of the AAAI-96 Workshop on Agent Modeling*, pages 46–52, Portland, OR.

Haynes, T. and Sen, S. (1997a). Crossover operators for evolving a team. In Koza, J. R., Deb, K., Dorigo, M., Fogel, D. B., Garzon, M., Iba, H., and Riolo, R. L., editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 162–167, Stanford University, CA, USA. Morgan Kaufmann.

Haynes, T., Sen, S., Schoenefeld, D., and Wainwright, R. (1995a). Evolving a team. In Siegel, E. V. and Koza, J. R., editors, *Working Notes for the AAAI Symposium on Genetic Programming*, pages 23–30, MIT, Cambridge, MA, USA. AAAI.

Haynes, T., Sen, S., Schoenefeld, D., and Wainwright, R. (1995b). Evolving multiagent coordination strategies with genetic programming. Technical Report UTULSA-MCS-95-04, The University of Tulsa.

Haynes, T., Wainwright, R., Sen, S., and Schoenefeld, D. (1995c). Strongly typed genetic programming in evolving cooperation strategies. In Eshelman, L., editor, *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, pages 271–278, Pittsburgh, PA, USA. Morgan Kaufmann.

Haynes, T. D. and Sen, S. (1997b). Co-adaptation in a team. *International Journal of Computational Intelligence and Organizations (IJCIO)*, 1(4).

Hölldobler, B. and Wilson, E. O. (1990). *The Ants*. Harvard University Press.

Hsu, W. H. and Gustafson, S. M. (2002). Genetic programming and multiagent layered learning by reinforcements. In Langdon, W. B., Cantú-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M., Schultz, A. C., Miller, J. F., Burke, E., and Jonoska, N., editors, *GECCO 2002: Proceedings of the Genetic*

and Evolutionary Computation Conference, pages 764–771, New York. Morgan Kaufmann Publishers.

Hu, J. and Wellman, M. (1996). Self-fulfilling bias in multiagent learning. In *Proceedings of the Second International Conference on Multi-Agent Systems*.

Hu, J. and Wellman, M. (1998a). Multiagent reinforcement learning: theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 242–250. Morgan Kaufmann, San Francisco, CA.

Hu, J. and Wellman, M. (1998b). Online learning about other agents in a dynamic multiagent system. In Sycara, K. P. and Wooldridge, M., editors, *Proceedings of the Second International Conference on Autonomous Agents (Agents'98)*, pages 239–246, New York. ACM Press.

Huang, J., Jennings, N. R., and Fox, J. (1995). An agent architecture for distributed medical care. In Wooldridge, M. and Jennings, N. R., editors, *Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890)*, pages 219–232. Springer-Verlag: Heidelberg, Germany.

Huhns, M. and Singh, M. (1998). Agents and multiagent systems: themes, approaches and challenges. In Huhns, M. and Singh, M., editors, *Readings in Agents*, pages 1–23. Morgan Kaufmann.

Huhns, M. and Weiß, G. (1998). Special issue on multiagent learning. *Machine Learning Journal*, 33(2-3).

Iba, H. (1996). Emergent cooperation for multiple agents using genetic programming. In Voigt, H.-M., Ebeling, W., Rechenberg, I., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature IV: Proceedings of the International Conference on Evolutionary Computation*, volume 1141 of *LNCS*, pages 32–41, Berlin, Germany. Springer Verlag.

Iba, H. (1998). Evolutionary learning of communicating agents. *Information Sciences*, 108(1–4):181–205.

Imam, I., editor (1996). *Intelligent Adaptive Agents. Papers from the 1996 AAAI Workshop. Technical Report WS-96-04*. AAAI Press.

Jennings, N., Sycara, K., and Wooldridge, M. (1998). A roadmap of agents research and development. *Autonomous Agents and Multi-Agent Systems*, 1:7–38.

Jennings, N., Varga, L., Aarnts, R., Fuchs, J., and Skarek, P. (1993). Transforming standalone expert systems into a community of cooperating agents. *International Journal of Engineering Applications of Artificial Intelligence*, 6(4):317–331.

Jim, K.-C. and Giles, C. L. (2000). Talking helps: Evolving communicating agents for the predator-prey pursuit problem. *Artificial Life*, 6(3):237–254.

Juille, H. and Pollack, J. (1998). Coevolving the "ideal" trainer: Application to the discovery of cellular automata rules. In *Proceedings of the Third Annual Genetic Programming Conference (GP-98)*.

Kapetanakis, S. and Kudenko, D. (2002a). Improving on the reinforcement learning of coordination in cooperative multi-agent systems. In *Proceedings of the Second Symposium on Adaptive Agents and Multi-agent Systems (AISB02)*.

Kapetanakis, S. and Kudenko, D. (2002b). Reinforcement learning of coordination in cooperative multi-agent systems. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI02)*.

Kendall, G. and Whitwell, G. (2001). An evolutionary approach for the tuning of a chess evaluation function using population dynamics. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC-2001)*, pages 995–1002. IEEE Press.

Kendall, G. and Willdig, M. (2001). An investigation of an adaptive poker player. In *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence (AI'01)*.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., and Osawa, E. (1997). RoboCup: The robot world cup initiative. In Johnson, W. L. and Hayes-Roth, B., editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 340–347, New York. ACM Press.

Lauer, M. and Riedmiller, M. (2000). An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 535–542. Morgan Kaufmann, San Francisco, CA.

Leerink, L. R., Schultz, S. R., and Jabri, M. A. (1995). A reinforcement learning exploration strategy based on ant foraging mechanisms. In *Proceedings of the Sixth Australian Conference on Neural Networks*, Sydney, Australia.

Lesser, V. (1999). Cooperative multiagent systems : A personal view of the state of the art. *IEEE Trans. on Knowledge and Data Engineering*, 11(1):133–142.

Lesser, V., Corkill, D., and Durfee, E. (1987). An update on the distributed vehicle monitoring testbed. Technical Report UM-CS-1987-111, University of Massachessets Amherst.

Lubberts, A. and Miikkulainen, R. (2001). Co-evolving a go-playing neural network. In *Coevolution: Turning Adaptive Algorithms upon Themselves, (Birds-on-a-Feather Workshop, Genetic and Evolutionary Computation Conference)*.

Luck, M., d'Inverno, M., Fisher, M., and FoMAS'97 Contributors (1998). Foundations of multi-agent systems: Techniques, tools and theory. *Knowledge Engineering Review*, 13(3):297–302.

Luke, S. (1998). Genetic programming produced competitive soccer softbot teams for RoboCup97. In Koza *et al*, J. R., editor, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 214–222. Morgan Kaufmann.

Luke, S., Hohn, C., Farris, J., Jackson, G., and Hendler, J. (1997). Co-evolving soccer softbot team coordination with genetic programming. In *Proceedings of the First International Workshop on RoboCup, at the International Joint Conference on Artificial Intelligence*, Nagoya, Japan.

Luke, S. and Spector, L. (1996). Evolving teamwork and coordination with genetic programming. In Koza, J. R., Goldberg, D. E., Fogel, D. B., and Riolo, R. L., editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 150–156, Stanford University, CA, USA. MIT Press.

Mahadevan, S. and Connell, J. (1991). Automatic programming of behavior-based robots using reinforcement learning. In *National Conference on Artificial Intelligence*, pages 768–773.

Mataric, M. (1994). *Interaction and Intelligent Behavior*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA. Also Technical Report AITR-1495.

Mataric, M. (1997). Reinforcement learning in the multi-robot domain. *Autonomous Robots*, 4(1):73–83.

Mataric, M., Nilsson, M., and Simsarian, K. (1995). Cooperative multi-robot box-pushing. In *Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems*, pages 556–561.

Miconi, T. (2001). A collective genetic algorithm. In Cantu-Paz *et al*, E., editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 876–883.

Miconi, T. (2003). When evolving populations is better than coevolving individuals: The blind mice problem. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*.

Mitchell, M., Crutchfield, J., and Das, R. (1996). Evolving cellular automata with genetic algorithms: A review of recent work. In *Proceedings of the First International Conference on Evolutionary Computation and its Applications (EvCA'96),*.

Mitchell, M., Crutchfield, J., and Hraber, P. (1994). Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, 75:361–391.

Monekosso, N., Remagnino, P., and Szarowicz, A. (2002). An improved Q-learning algorithm using synthetic pheromones. In B. Dunin-Keplicz, E. N., editor, *From Theory to Practice in Multi-Agent Systems, Second International Workshop of Central and Eastern Europe on Multi-Agent Systems, CEEMAS 2001 Cracow, Poland, September 26-29, 2001. Revised Papers*, Lecture Notes in Artificial Intelligence LNAI-2296. Springer-Verlag.

Monekosso, N. D. and Remagnino, P. (2001). Phe-Q: A pheromone based Q-learning. In *Australian Joint Conference on Artificial Intelligence*, pages 345–355.

Monekosso, N. D. and Remagnino, P. (2002). An analysis of the pheromone Q-learning algorithm. In *Proceedings of the VIII Iberoamerican Conference on Artificial Intelligence IBERAMIA-02*, pages 224–232.

Monekosso, N. D., Remagnino, P., and Szarowicz, A. (2001). An improved Q-learning algorithm using synthetic pheromones. In *Proceedings of the Second Workshop of Central and Eastern Europe on Multi-Agent Systems CEEMAS-01*, pages 197–206.

Mori, J., Torikoshi, H., Nakai, K., Mori, K., and Masuda, T. (1988). Computer control system for iron and steel plants. *Hitachi Review*, 37(4):251–258.

Morley, R. E. and Schelberg, C. (1993). An analysis of a plant-specific dynamic scheduler. In *Proceedings of the NSF Workshop on Dynamic Scheduling*.

Mukherjee, R. and Sen, S. (2001). Towards a pareto-optimal solution in general-sum games. In *Agents-2001 Workshop on Learning Agents*.

Mukhopadjyay, U., Stephens, L., and Huhns, M. (1986). An intelligent system for document retrieval in distributed office environments. *Journal of the American Society for Information Science*, 37(3):123–135.

Muller, J. and Pischel, M. (1994). An architecture for dynamically interacting agents. *Journal of Intelligent and Cooperative Information Systems*, 3(1):25–45.

Mundhe, M. and Sen, S. (2000a). Evaluating concurrent reinforcement learners. In *Proceedings of the International Conference on Multiagent System*.

Mundhe, M. and Sen, S. (2000b). Evolving agent societies that avoid social dilemmas. In Whitley, D., Goldberg, D., Cantu-Paz, E., Spector, L., Parmee, I., and Beyer, H.-G., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 809–816, Las Vegas, Nevada, USA. Morgan Kaufmann.

Nagayuki, Y., Ishii, S., and Doya, K. (2000). Multi-agent reinforcement learning: An approach based on the other agent's internal model. In *Proceedings of the International Conference on Multi-Agent Systems (ICMAS-00)*.

Nagendra-Prasad, M. V. (1997). *Learning Situation-Specific Control in Multi-Agent Systems*. PhD thesis, University of Massachusetts Amherst.

Ohko, T., Hiraki, K., and Arzai, Y. (1997). Addressee learning and message interception for communication load reduction in multiple robots environments. In Weiß, G., editor, *Distributed Artificial Intelligence Meets Machine Learning : Learning in Multi-Agent Environments, Lecture Notes in Artificial Intelligence 1221*. Springer-Verlag.

Ostergaard, E., Sukhatme, G., and Mataric, M. (2001). Emergent bucket brigading - a simple mechanism for improving performance in multi-robot constrainedspace foraging tasks. In *Proceedings of the Fifth International Conference on Autonomous Agents,*.

Pagie, L. and Mitchell, M. (2001). A comparison of evolutionary and coevolutionary search. In Belew, R. K. and Juillè, H., editors, *Coevolution: Turning Adaptive Algorithms upon Themselves*, pages 20–25, San Francisco, California, USA.

Panait, L. and Luke, S. (2004a). Ant foraging revisited. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*.

Panait, L. and Luke, S. (2004b). Learning ant foraging behaviors. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*.

Panait, L. A., Wiegand, R. P., and Luke, S. (2003). Improving coevolutionary search for optimal multiagent behaviors. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*.

Parker, L. (2000a). Current state of the art in distributed autonomous mobile robotics. In Parker, L., Bekey, G., and Barhen, J., editors, *Distributed Autonomous Robotic Systems 4,*, pages 3–12. Springer-Verlag.

Parker, L. (2000b). Multi-robot learning in a cooperative observation task. In *Proceedings of Fifth International Symposium on Distributed Autonomous Robotic Systems (DARS 2000)*.

Parker, L. (2002). Distributed algorithms for multi-robot observation of multiple moving targets. *Autonomous Robots*, 12(3).

Parker, L., Touzet, C., and Fernandez, F. (2001). Techniques for learning in multi-robot teams. In Balch, T. and Parker, L., editors, *Robot Teams: From Diversity to Polymorphism*. AK Peters.

Peceny, M., Weiß, G., and Brauer, W. (1996). Verteiltes maschinelles lernen in fertigungsumgebungen. Technical Report FKI-218-96, Institut fur Informatik, Technische Universitat Munchen.

Pollack, J. and Blair, A. (1998). Coevolution in the successful learning of backgammon strategy. *Machine Learning*, 32(3):225–240.

Pollack, J., Blair, A., and Land, M. (1997). Coevolution of a backgammon player. In Langton, C. G. and Shimohara, K., editors, *Artificial Life V: Proc. of the Fifth Int. Workshop on the Synthesis and Simulation of Living Systems*, pages 92–98, Cambridge, MA. The MIT Press.

Potter, M. (1997). *The Design and Analysis of a Computational Model of Cooperative Coevolution*. PhD thesis, George Mason University, Fairfax, Virginia.

Potter, M. and De Jong, K. (1994). A cooperative coevolutionary approach to function optimization. In Davidor, Y. and Schwefel, H.-P., editors, *Proceedings of the Third International Conference on Parallel Problem Solving from Nature (PPSN III)*, pages 249–257. Springer-Verlag.

Potter, M. and De Jong, K. (2000). Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29.

Potter, M., De Jong, K., and Grefenstette, J. J. (1995). A coevolutionary approach to learning sequential decision rules. In *Proceedings from the Sixth International Conference on Genetic Algorithms*, pages 366–372. Morgan Kaufmann Publishers, Inc.

Potter, M., Meeden, L., and Schultz, A. (2001). Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *Proceedings of The Seventeenth International Conference on Artificial Intelligence (IJCAI-2001)*.

Quinn, M. (2001a). A comparison of approaches to the evolution of homogeneous multi-robot teams. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC2001)*, pages 128–135, COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea. IEEE Press.

Quinn, M. (2001b). Evolving communication without dedicated communication channels. In *Advances in Artificial Life: Sixth European Conference on Artificial Life (ECAL01)*.

Quinn, M., Smith, L., Mayley, G., and Husbands, P. (2002). Evolving formation movement for a homogeneous multi-robot system: Teamwork and role-allocation with real robots. Cognitive Science Research Paper 515. School of Cognitive and Computing Sciences, University of Sussex, Brighton, BN1 9QG. ISSN 1350-3162.

Reynolds, C. (1994). Competition, coevolution and the game of tag. In Brooks, R. A. and Maes, P., editors, *Artificial Life IV, Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems.*, pages 59–69. MIT Press.

Reynolds, C. W. (1987). Flocks, herds, and schools: a distributed behavioral model. *Computer Graphics*, 21(4):25–34.

Salustowicz, R., Wiering, M., and Schmidhuber, J. (1997). Learning team strategies with multiple policy-sharing agents: A soccer case study. Technical report, ISDIA, Corso Elvezia 36, 6900 Lugano, Switzerland.

Salustowicz, R., Wiering, M., and Schmidhuber, J. (1998). Learning team strategies: Soccer case studies. *Machine Learning*, 33(2/3):263–282.

Samuel, A. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229.

Saunders, G. and Pollack, J. (1996). The evolution of communication schemes over continuous channels. In *From Animals to Animats 4 - Proceedings of the Fourth International Conference on Adaptive Behaviour,*.

Sauter, J., Matthews, R. S., Van Dyke Parunak, H., and Brueckner, S. (2002). Evolving adaptive pheromone path planning mechanisms. In *Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, pages 434–440.

Sauter, J., Van Dyke Parunak, H., Brueckner, S., and Matthews, R. (2001). Tuning synthetic pheromones with evolutionary computing. In Smith, R. E., Bonacina, C., Hoile, C., and Marrow, P., editors, *Evolutionary Computation and Multi-Agent Systems (ECOMAS)*, pages 321–324, San Francisco, California, USA.

Schmidhuber, J. and Zhao, J. (1996). Multi-agent learning with the success-story algorithm. In *ECAI Workshop LDAIS / ICMAS Workshop LIOME*, pages 82–93.

Schultz, A., J.Grefenstette, and Adams, W. (1996). Robo-shepherd: Learning complex robotic behaviors. In *Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*, pages 763–768. ASME Press.

Schwuttke, U. M. and Quan., A. G. (1993). Enhancing performance of cooperating agents in realtime diagnostic systems. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*.

Sen, S., editor (1996). *Adaptation, Coevolution and Learning in Multiagent Systems. Papers from the 1996 AAAI Spring Symposium. Technical Report SS-96-01.* AAAI Press.

Sen, S. (1997). Multiagent systems: Milestones and new horizons. *Trends in Cognitive Science*, 1(9):334–339.

Sen, S. (1998). Special issue on evolution and learning in multiagent systems. *International Journal of Human-Computer Studies*, 48(1).

Sen, S. and Sekaran, M. (1996). Multiagent coordination with learning classifier systems. In Weiß, G. and Sen, S., editors, *Proceedings of the IJCAI Workshop on Adaption and Learning in Multi-Agent Systems*, volume 1042, pages 218–233. Springer Verlag.

Sen, S. and Sekaran, M. (1998). Individual learning of coordination knowledge. *Journal of Experimental and Theoretical Artificial Intelligence*, 10(3):333–356.

Sen, S., Sekaran, M., and Hale, J. (1994). Learning to coordinate without sharing information. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 426–431.

Singh, S. (1992). Transfer of learning across sequential tasks. *Machine Learning*, 8:323–339.

Smith, R. and Gray, B. (1993). Co-adaptive genetic algorithms: An example in othello strategy. Technical Report TCGA 94002, University of Alabama, Department of Engineering Science and Mechanics.

Steeb, R., Cammarata, S., Hayes-Roth, F., Thorndyke, P., and Wesson, R. (1988). Distributed intelligence for air fleet control. In Bond, A. and Gasser, L., editors, *Readings in Distributed Artificial Intelligence*, pages 90–101. Morgan Kaufmann Publishers.

Steels, L. (1995). A self-organizing spatial vocabulary. *Artificial Life*, 2(3):319–332.

Steels, L. (1996a). Emergent adaptive lexicons. In Maes, P., editor, *Proceedings of the Simulation of Adaptive Behavior Conference*. MIT Press.

Steels, L. (1996b). Self-organising vocabularies. In *Proceedings of Artificial Life V*.

Steels, L. (1996c). The spontaneous self-organization of an adaptive language. In Muggleton, S., editor, *Machine Intelligence 15*. Oxford University Press, Oxford, UK.

Steels, L. (1997). Synthesising the origins of language and meaning using co-evolution, self-organisation and level formation. In Hurford, J., Knight, C., and Studdert-Kennedy, M., editors, *Approaches to the Evolution of Language: Social and Cognitive bases*. Edinburgh University Press.

Steels, L. (2000). The puzzle of language evolution. *Kognitionswissenschaft*, 8(4):143–150.

Steels, L. and Kaplan, F. (1999). Collective learning and semiotic dynamics. In *Proceedings of the European Conference on Artificial Life*, pages 679–688.

Stone, P. (1997). Layered learning in multiagent systems. In *Proceedings of National Conference on Artificial IntelligenceAAAI/IAAI*.

Stone, P. (1998). *Layered Learning in Multi-Agent Systems*. PhD thesis, Carnegie Mellon University.

Stone, P. and Sutton, R. (2002). Keepaway soccer: A machine learning testbed. In Birk, A., Coradeschi, S., and Tadokoro, S., editors, *RoboCup 2001: Robot Soccer World Cup V*, volume 2377 of *Lecture Notes in Computer Science*, pages 214–223. Springer.

Stone, P. and Veloso, M. M. (2000). Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383.

Suematsu, N. and Hayashi, A. (2002). A multiagent reinforcement learning algorithm using extended optimal response. In *Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, pages 370–377.

Suryadi, D. and Gmytrasiewicz, P. J. (1999). Learning models of other agents using influence diagrams. In *Preceedings of the 1999 International Conference on User Modeling*, pages 223–232.

Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative learning. In Huhns, M. N. and Singh, M. P., editors, *Readings in Agents*, pages 487–494. Morgan Kaufmann, San Francisco, CA, USA.

Tangamchit, P., Dolan, J., and Khosla, P. (2002). The necessity of average rewards in cooperative multirobot learning. In *Proceedings of IEEE Conference on Robotics and Automation*.

Van Dyke Parunak, H. (1996). Applications of distributed artificial intelligence in industry. In O'Hare, G. M. P. and Jennings, N. R., editors, *Foundations of Distributed AI*. John Wiley & Sons.

Varga, L. Z., Jennings, N. R., and Cockburn, D. (1994). Integrating intelligent systems into a cooperating community for electricity distribution management. *International Journal of Expert Systems with Applications*, 7(4):563–579.

Vidal, J. and Durfee, E. (1997). Agents learning about agents: A framework and analysis. In *Working Notes of AAAI-97 Workshop on Multiagent Learning*.

Weihmayer, R. and Velthuijsen, H. (1994). Application of distributed AI and cooperative problem solving to telecommunications. In Liebowitz, J. and Prereau, D., editors, *AI Approaches to Telecommunications and Network Management*. IOS Press.

Weiß, G. (1994). Some studies in distributed machine learning and organizational design. Technical Report FKI-189-94, Institut für Informatik, TU München.

Weiß, G. (1995). *Distributed Machine Learning*. Sankt Augustin: Infix Verlag.

Weiß, G., editor (1997). *Distributed Artificial Intelligence Meets Machine Learning : Learning in Multi-Agent Environments*. Number 1221 in Lecture Notes in Artificial Intelligence. Springer-Verlag.

Weiß, G. (1998). Special issue on learning in distributed artificial intelligence systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 10(3).

Weiß, G., editor (1999). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press.

Weiß, G. and Dillenbourg, P. (1999). What is 'multi' in multi-agent learning? In Dillenbourg, P., editor, *Collaborative learning. Cognitive and computational approaches*, pages 64–80. Pergamon Press.

Weiß, G. and Sen, S., editors (1996). *Adaptation and Learning in Multiagent Systems*. Lecture Notes in Artificial Intelligence, Volume 1042. Springer-Verlag.

Wellman, M. and Hu, J. (1998). Conjectural equilibrium in multiagent learning. *Machine Learning*, 33(2-3):179–200.

Werfel, J., Mitchell, M., and Crutchfield, J. P. (2000). Resource sharing and coevolution in evolving cellular automata. *IEEE Transactions on Evolutionary Computation*, 4(4):388.

White, T., Pagurek, B., and Oppacher, F. (1998). ASGA: Improving the ant system by integration with genetic algorithms. In Koza, J. R., Banzhaf, W., Chellapilla, K., Deb, K., Dorigo, M., Fogel, D. B., Garzon, M. H., Goldberg, D. E., Iba, H., and Riolo, R., editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 610–617, University of Wisconsin, Madison, Wisconsin, USA. Morgan Kaufmann.

Wiegand, R. P. (1998). Applying diffusion to a cooperative coevolutionary model. In Eiben, A. E., Baeck, T., Schoenauer, M., and Schwefel, H.-P., editors, *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature (PPSN V)*, pages 560–569. Springer-Verlag.

Wiegand, R. P., Liles, W., and De Jong, K. (2001). An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In Cantu-Paz *et al*, E., editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 1235–1242.

Wiegand, R. P., Liles, W., and De Jong, K. (2002a). Analyzing cooperative coevolution with evolutionary game theory. In Fogel, D., editor, *Proceedings of Congress on Evolutionary Computation (CEC-02)*, pages 1600–1605. IEEE Press.

Wiegand, R. P., Liles, W., and De Jong, K. (2002b). Modeling variation in cooperative coevolution using evolutionary game theory. In Poli, R., Rowe, J., and Jong, K. D., editors, *Foundations of Genetic Algorithms (FOGA) VII*, pages 231–248. Morgan Kaufmann.

Wiering, M., Salustowicz, R., and Schmidhuber, J. (1999). Reinforcement learning soccer teams with incomplete world models. *Journal of Autonomous Robots*, 7(1):77–88.

Wilson, E. (1975). *Sociobiology: The New Synthesis*. Belknap Press.

Wooldridge, M., Bussmann, S., and Klosterberg, M. (1996). Production sequencing as negotiation. In *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM-96),*.

Wooldridge, M. and Jennings, N. (1995). Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152.

Yanco, H. and Stein, L. (1993). An adaptive communication protocol for cooperating mobile robots. In *From Animals to Animats: International Conference on Simulation of Adaptive Behavior*, pages 478–485.

Zhang, B. and Cho, D. (1998). Coevolutionary fitness switching: Learning complex collective behaviors using genetic programming. In *Advances in Genetic Programming III*, pages 425–445. MIT Press.

Zhao, Q. (1998). A general framework for cooperative co-evolutionary algorithms: a society model. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, pages 57–62.