Department of Computer Science
George Mason University
Technical Report Series

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/
703-993-1530

Cooperative Multi-Agent Learning: The State of the Art

Liviu Panait lpanait@cs.gmu.edu
Sean Luke sean@cs.gmu.edu

Technical Report GMU-CS-TR-2003-1

Abstract

Cooperative multi-agent systems problems are ones in which several agents attempt, through their interaction, to jointly solve tasks or to maximize their utility. Due to the interactions among the agents, multi-agent problem complexity can rise rapidly with the number of agents or their behavioral sophistication. The challenge this presents to the task of programming solutions to such problems has spawned increasing interest in machine learning (ML) techniques to automate the search and optimization process.

We provide a broad survey of the cooperative multi-agent learning literature. Previous surveys of this area have largely focused on issues common to specific subareas (for example, reinforcement learning or robotics). In this survey we attempt to draw from multi-agent learning work in a spectrum of areas, including reinforcement learning, evolutionary computation, game theory, complex systems, agent modeling, and robotics.

We find that this broad view leads to a division of the work into two categories, each with its own special issues: applying a single learner to discover joint solutions to multi-agent problems (*team learning*), or using multiple simultaneous learners, often one per agent (*concurrent learning*). Additionally, we discuss two important topics independent of these categories: problem decomposition and communication. We conclude with a presentation of multi-agent learning problem domains, a discussion of certain challenge topics (scalability and adaptive dynamics), and a list of multi-agent learning resources.

1 Introduction

In recent years there has been increased interest in decentralized approaches to solving complex real-world problems. Most such approaches fall into the area of *distributed systems*, where a number of entities work together to cooperatively solve problems. The combination of distributed systems and artificial intelligence (AI) is collectively known as *distributed artificial intelligence* (DAI). Traditionally, DAI is divided into two areas. The first area, *distributed problem solving*, is usually concerned with the decomposition and distribution of a problem solving process among multiple slave nodes, and the collective construction of a solution to the problem. The second class of approaches, *multi-agent systems* (MAS), emphasizes the joint behaviors of agents with some degree of autonomy.

In this survey, we will focus on the application of *machine learning* (ML) to problems in the MAS area. Machine learning explores ways to automate the inductive process: getting a machine agent to discover on its own, through repeated trials, how to solve a given task or to minimize error. Machine learning has proven a popular approach to solving multi-agent systems problems because the inherent complexity of many such problems can make solutions by hand prohibitively difficult. Automation is attractive. We will additionally focus on problem domains in which the multiple agents are *cooperating* to solve a joint task or to maximize utility; as opposed to *competing* with one another. We call this specific subdomain of interest *cooperative multi-agent learning*. Despite the relative youth of the field, the number of multi-agent learning papers is large, and we hope that this survey will prove helpful in navigating the current body of work.

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 multi-agent systems according to [248, 106]

1.1 Multi-Agent Systems

The terms *agent* and *multi-agent* are not well-defined in the community; we offer our own, admittedly broad, definitions of the concepts here as we intend to use them later in the survey. An *agent* is a computational mechanism that exhibits a high degree of autonomy, performing actions in its environment based on information (sensors, feedback) received from the environment. A *multi-agent* environment is one in which there is more than one agent, and further, there are constraints on that environment such that agents may not at any given time know *everything* about the world that other agents know (including the internal states of the other agents themselves). Otherwise, the multi-agent system's dynamics may be analogous to a single-agent system where each of the "agents" is really an appendage of a master controller.

Before continuing, we mention few other definitions for agents and multi-agent problems in the literature. According to Wooldridge and Jennings [267], there are two general usages of the term agent: *weak agents*, which often exhibit autonomy, social ability, reactivity, interaction via a communication language, pro-activeness, and the ability to perceive the environment and respond to it accordingly; and *strong agents*, which may have beliefs, desires, intentions, knowledge, commitments, and other human-like characteristics. Jennings et al. [114] suggest that most multi-agent systems applications feature agents with incomplete information about the environment, a lack of centralized control, decentralized and distributed information, and asynchronous computation.

Depending on their interest, several authors have provided different taxonomies for MAS applications. For example, Dudek et al. [65] classify swarm robotics applications according to team size, range, communication topology and bandwidth, team composition and reconfigurability, and the processing ability of individual agents. In a collection describing the application of Distributed Artificial Intelligence to industry, Parunak [171] differentiates between *agent characteristics* (team heterogeneity, control architectures, input/output abilities) and *system characteristics* (for example, communication settings). Stone [223] and Stone and Veloso [225] explicitly distinguish among four groups divided by heterogeneity vs. homogeneity and by communication versus lack thereof. Last, Weiß [246, 248] and Huhns and Singh [106] define a number of multi-agent systems characteristics, summarized in Table 1.

1.2 Multi-Agent Learning

Much of the multi-agent learning literature has sprung from historically somewhat separate communities — notably reinforcement learning and dynamic programming, robotics, evolutionary computation, and complex systems. Existing surveys of the work have likewise tended to emphasize issues special to only some subset of these areas. As there is increasing cross-fertilization among these fields, we believe a new, unified survey of multi-agent learning may be useful to provide a joint overview of the state of the art.

Accordingly, we begin the survey by defining multi-agent learning broadly: it is the application of machine learning to problems involving multiple agents. We think that there are two features of multi-agent learning which merit its study as a field separate from ordinary machine learning. First, because multi-agent 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 unpredictable changes in the resulting macro-level ("emergent") properties of the multi-agent group as a whole. Second, multi-agent 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.[1]

[1]Our argument here differs somewhat from Weiß and Dillenbourg [249], who believe that what makes multi-agent learning special are: separate learning algorithms, task decomposition, agent interaction, conflict resolution among agents, mutual regulation among the agents,

A large portion of the multi-agent literature may be neatly divided according to whether or not it involves the second feature (multiple learners); and this division forms the top level of our survey taxonomy. Some literature applies a single learner 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 team as a whole. Other literature applies separate learning processes to individual agents, while still assessing the quality of the agents as a team. We term this approach *Concurrent Learning*.

To illustrate the difference between the two, consider a problem domain involving a room with two robots and a box. The robots need to cooperatively push the box from its initial location to a target location, and the only performance measure provided how close to the target location that *both robots together* have pushed the box. A Team Learning approach employs a single learner to iteratively improve the "team behavior" – which consists of *both* robot behaviors – according to the performance measure provided as feedback. The Concurrent Learning approach allows each agent (robot) to modify its own behavior via the agent's *own* learning process. The performance measure now must be apportioned among the two agents (perhaps by dividing it equally). The agents will improve their behaviors independent of one another, but have little or no control over how the other agents decide to behave.

Research in Concurrent Learning has broken down along different lines than that in Team Learning, primarily because of differences in the dynamics of the techniques. As Team Learning uses a single learner, most research has focused on the *representation* of candidate solutions that the learner is developing: in particular, the degree of heterogeneity among the team members. More heterogeneity leads to more specialization and to more sophisticated team behaviors, but also increases the search space. In contrast, Concurrent Learning literature has largely focused on the relationships the learners have with one another: the game-theoretic properties of the agents' interactions, the apportioning of the performance measure among them, and how agents may model one another.

Additionally, there are a two large research areas which are somewhat independent of the learning process: communication and problem decomposition. We discuss these areas in their own sections. We conclude the survey with a large collection of multi-agent problem domains drawn from the gamut of the literature, some challenge issues (scalability and adaptive dynamics) plus a list of existing resources. The survey follows the following format:

and the ability of agents to explain the reasoning to one another.

Multi-Agent Learning Categories
- ***Team Learning***
  - Homogeneous Team Learning approaches
  - Heterogeneous Team Learning approaches
  - Hybrid Team Learning approaches
- ***Concurrent Learning***
  - Dynamics of learning
  - Credit assignment and locality of reinforcement
  - Concurrent modeling

Category-independent Research Areas
- ***Problem Decomposition***
- ***Communication and Learning***
  - Direct Communication
  - Indirect Communication

Challenges
- ***Scalability***
- ***Adaptive Dynamics***

Problem Domains
- ***Embodied Agents***
- ***Game-Theoretic Environments***
- ***Real-World Applications***

Resources and Conclusions

We believe that this taxonomy provides a good arrangement of multi-agent learning research. However, there are a few issues which recur several times during the survey and are worth mentioning here:

Credit Assignment. When evaluating an agent, an important problem that needs to be solved is that of deciding which one of a series of agent actions led to the reinforcement received from the environment. Additionally, when the reinforcement is attributed to a team's behavior, assessing the credit an agent receives, and its share of the team reinforcement, can also be very important to the learning process. The two subproblems are known as the *intra-agent* and *extra-agent* credit assignment problems [250].

Game-theoretic Problems. An important related issue is how to cast a given multi-agent problem as a cooperative problem (as opposed to a competitive one, etc.). This is not a trivial problem: many multi-agent 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 *indifferent* to one another. There are

of course many game-theoretic environments which do not fall into any of these categories.

Co-adaptation. Another interesting, game-theoretic multi-agent learning issue concerns the fact that the learning processes are not independent, but they affect each other. Consider when 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" for 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, and may gravitate towards balanced equilibrium rather than to an optimum.

The survey is mainly concerned with cooperative multi-agent learning, rather than competitive learning methods. However, it is worth mentioning one important class of competitive multi-agent 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 [194]. However, the bulk of learned game-playing work has been relatively recent. For example, Luke [133] evolved soccer-playing softbot teams, and Fogel [73] evolved a highly human-competitive checkers program called Blondie24. Tesauro's TD-Gammon program ranked among the best players in the world in the game of backgammon [233]. Other investigations concerned games such as Tic-Tac-Toe [5], Backgammon [175, 174], Mancala [56], Othello [211], pursuit-evasion [48, 88], Go [131], Chess [234, 121], Poker [122], Blackjack [229] and Tag [187].

Another large class of learning in multi-agent systems that we will ignore in this survey involves situations where a single agent learns while the other agents' behaviors are fixed. One of the many examples of such learning investigations is presented in [83]. 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.

1.3 Machine Learning Methods

There are three main approaches to learning: *supervised*, *unsupervised*, and *reward-based*[2] learning. These methods are distinguished by what kind of feedback the critic provides to the learner. In supervised learning, the critic provides the correct output. In unsupervised learning, no feedback is provided at all. In reward-based learning, the critic provides a quality assessment (the "reward") of the learner's output.

Because of the inherent complexity in the interactions of multiple agents, various machine learning methods — notably supervised learning methods — are not easily applied to the problem because they typically assume a critic which can provide the agents with the "correct" behavior for a given situation (a notable exception involving teaching in the context of mutual supervised learners is presented in [78]). Thus the large majority of papers in this field have used reward-based methods. The reward-based learning literature may be approximately divided into two subsets: *reinforcement learning* methods which estimate value functions; and *stochastic search* methods such as evolutionary computation, simulated annealing, and stochastic hill-climbing, which directly learn behaviors without appealing to value functions. In the stochastic search literature, most multi-agent discussion concentrates on evolutionary computation.

Reinforcement Learning Reinforcement learning (RL) methods are particularly useful in domains where reinforcement[3] information (expressed as penalties or rewards) is provided after a sequence of actions performed in the environment. Q-Learning and Temporal-Difference (TD(λ)) Learning are two common RL methods; the former learns the utility of performing actions in states, while the latter usually learns the utility of being in the states themselves. Reinforcement learning methods are inspired by dynamic programming concepts and require formulas for updating the expected utilities and for using them for the exploration of the state space. The update is often a weighted sum of the current value, the reinforcement obtained when performing an action, and the expected utility of the next state reached after the action is performed. While exploring, deterministic strategies may choose the most promising action to be performed in each state (according to the expected utilities). Other stochastic techniques may lead to better exploration of the state space. Reinforcement learning methods have theoretical proofs of convergence; unfortunately, such convergence assumptions do not hold for some real-world applications. For more information on reinforcement learning techniques, [230, 14, 118] are good starting points.

[2] Some of the literature calls this "reinforcement learning"; but to avoid confusion with a specific subset *also* commonly called reinforcement learning (including techniques such as Q Learning), we will use *reward-based* to distinguish the superset from the subset. The term "reward" is admittedly problematic though, as there exist "negative rewards" (punishments).

[3] In this survey, we use the terms *reward* and *reinforcement* interchangeably to denote the information the agents receive from the environment as a consequence of their actions.

Evolutionary Computation Evolutionary Computation (EC) (or *Evolutionary Algorithms* (EAs)) is a family of techniques in which abstract Darwinian models of evolution are applied to refine populations of candidate solutions (known as "individuals") to a given problem. An evolutionary algorithm begins with an initial population of randomly-generated individuals. Each member of this population is then evaluated and assigned a fitness (a quality assessment). The EA then uses a fitness-oriented procedure to select, breed, and mutate individuals to produce children which are then added to the population, replacing older individuals. One evaluation, selection, and breeding cycle is known as a *generation*. Successive generations continue to refine the population until time is exhausted or a sufficiently fit individual is discovered. Evolutionary computation methods include *genetic algorithms* (GA) and *evolution strategies* (ES), which are usually applied to the search of multidimensional parameter spaces, and *genetic programming* (GP), which concerns itself with evolving actual computer programs. There are many sources of additional information on EC; good choices include [99, 77, 124, 9, 144, 74, 58, 59].

Coevolutionary algorithms (CEAs) represent a natural approach to applying evolutionary computation to refine multi-agent behaviors. In a CEA, the fitness of an individual is based on its interaction with other individuals in the population: thus the fitness assessment is context-sensitive and subjective. In *competitive* coevolution, individuals benefit at the expense of their peers; but in *cooperative* coevolution, individuals succeed or fail together in collaboration. A standard approach to applying cooperative coevolutionary algorithms (or CCEAs) to an optimization problem starts by decomposing the problem representation into subcomponents, then assigning each subcomponent to a separate population of individuals [181, 178, 177, 179].

2 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 can be unexpected: this notion is often dubbed the *emergent complexity* of the multi-agent system.

Team learning is an easy approach to multi-agent 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 concurrent 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 concurrent learning. Finally, 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.[4]

Team learning has some disadvantages as well. A major problem with team learning is the increasingly 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) [192, 193, 208, 110].

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.

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 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.

2.1 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 search

[4] 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, then this is equivalent to a multiple-learner (concurrent learning) situation. We have chosen to put such gray-area models in the team learning category.

space is simply too large to use heterogeneous learning, even if heterogeneity would ultimately yield the best results. A large number of homogeneous team learning papers are concerned with communication issues; we discuss such literature in Section 5.

Haynes et al. [96, 95, 97], Haynes and Sen [90], Haynes et al. [89] 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 with the best human-coded greedy predator algorithms, both when using and when not using information on the position of the other predators. However, when coevolving the prey and predator behaviors, the genetic programming system employed discovers a prey that evades all previously reported hand-coded, greedy, and evolved predators. The authors suggest that better performance may be obtainable with communicating agents. Jim and Giles [116] 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 performance superior to all previously reported work in this domain.

Quinn et al. [185] investigate the use of evolutionary computation techniques for a team formation problem. Three agents start from random positions but close enough to sense one another. 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 perform better as a team.

Salustowicz et al. [192, 193] compare PIPE and CO-PIPE (population-based algorithms similar to 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 for a value function. On the other hand, both PIPE and CO-PIPE search directly in the policy space and show good performance. The authors conclude that searching in policy space may be preferable in other multi-agent domains as well. A similar result is reported in [208], where a "Bucket-Brigade" evolutionary algorithm proves competitive with Q-learning in a coordinated navigation problem.[5] A contradicting result is reported in [261], where a modified Q-learning outperforms the methods earlier reported in [192, 193].

Cellular Automata A cellular automaton (CA) is an oft-overlooked paradigm for homogeneous team learning[6]. A CA consists of a neighborhood (often a row or grid) of agents, each with its own internal state, plus a state-update agent behavior (the *rule*) applied to all the agents synchronously. This rule is usually based on the current states of an agent's neighbors. CAs have many of the hallmarks of a multi-agent system: iteractions and communications are local, and behaviors are performed independently. A good survey of existing work in learning cellular automata rules is presented in [147].

One common CA problem, the Majority Classification[7] task, asks for an update rule which — given initial configurations of agents, each agent with an internal state of 1 or 0 — classifies correctly as many of them as possible based on whether the initial configuration had 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 (similarly for 0's). If it has not converged, the rule has not classified the initial configuration. The goal is to discover a rule which classifies most configurations correctly given specific standard settings (in terms of number of agents, size of neighborhood, etc). Due to the complexity of the emergent behavior, it is exceptionally difficult to create good performing solutions for this problem by hand. The best human-coded result, of 82.178% accuracy, was proposed by Das et al. [55]. Much better results have been produced with evolutionary computation methods: Andre et al. [3] report a rule with accuracy 82.326% using genetic programming. Several authors, including Werfel et al. [252], Pagie and Mitchell [163], Juille and Pollack [117], suggest that coevolution[8] might perform well in this problem domain; at present the best known result, with accuracy 86.3%, was obtained via coevolution [117].

[5] This paper is a concurrent learning paper, not a team learning one, but we place it here as it supports the conclusions of Salustowicz et al. [192, 193].

[6] There are a few CA papers examining heterogeneous agents.

[7] Also known as Density Classification.

[8] The papers refer to a particular case of coevolution, namely competitive coevolution, and a special setting containing two subpopulations. The first subpopulation consists of candidate solutions for the problem (in this case, behaviors for the CA agents). The second subpopulation contains possible test cases – initial configurations that the candidate solutions should solve. Candidate solutions in the first subpopulation are considered better fit when solving more test cases existing in the second subpopulation; meanwhile, the fitness of a test case is based on how many candidate solutions it stumps.

2.2 Heterogeneous Team Learning

In Heterogeneous Team Learning, the team is composed of agents with different behaviors, with a single learner trying to improve the team as a whole. This approach allows for more diversity in the team at the cost of increasing the search space. The bulk of research in Heterogeneous Team Learning has concerned itself with the requirement for or the emergence of specialists.

When are specialists really needed in a team? Balch [11][9] suggests that domains where single agents can perform well (for example, foraging) are particularly suited for homogeneous learning, while domains that require task specialization (such as robotic soccer) are more suitable for heterogeneous approaches. His argument is bolstered by Bongard [22],[9] who hypothesizes that heterogeneity may be better in inherently decomposable domains. Potter et al. [180][9] suggest that domain difficulty is not a determinant factor requiring a heterogeneous approach. They experiment with increasingly difficult versions of a multi-agent herding domain obtained by adding predators. Potter et al. argue that increasing the number of different skills required to solve the domain is a determinant factor.

Luke and Spector [135] investigate possible alternatives for evolving teams of agents. In their experiments, Luke and Spector compare homogeneity, heterogeneity with restricted breeding (agents could only breed with like agents from other teams), and heterogeneity with no breeding restrictions. The authors suggest that the restricted breeding works better than having no restrictions 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. Similarly, Andre and Teller [4] apply genetic programming to develop a team of soccer playing agents for the RoboCup simulator. The individuals encode eleven different behaviors (one for each player). Andre and Teller mention that the crossover operator (between teams of agents) was most successful when performing restricted breeding.

Haynes and Sen [92, 98, 94] investigate the evolution of homogeneous and heterogeneous teams for the predator-prey pursuit domain. The authors present several crossover operators that may promote the appearance of specialists within the teams. The results indicate that team heterogeneity can significantly help despite 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 location. Haynes and Sen report that the most successful crossover operator (TeamUniform) allows arbitrary crossover operations between behaviors for different agents, a result contradicting Luke and Spector [135] and Andre and Teller [4].

Another application of coevolution to team learning treats each individual in the population not as a team of agents but as a single agent; agents in the population are evaluated by grouping them together to form a team [41, 183]. Quinn shows a situation where this method outperforms a homogeneous team learning approach. Similarly, Miconi [145] evaluates an individual by forming a team from the population, then comparing how much worse the team performs when the indivdiual is *not* in it. Miconi reports the appearance of squads and "subspecies" in this method.

2.3 Hybrid Team Learning

Luke [133], Luke et al. [134] 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 each, and then evolves six behaviors, one per squad. 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 given more time.

Hara and Nagao [87] present an innovative method for hybrid group learning. Faced with the specialization/search-space tradeoff inherent in heterogeneity, the authors suggest an automated grouping technique called Automatically Defined Groups (ADG). They apply it successfully to a simple load transportation problem and to a modified tile-world domain. In ADG, the team of agents is composed of several groups of homogeneous agents (similar to [133, 134]); however, ADG automatically discovers the optimum number of groups and their compositions. Additionally, the acquired group structure may give insights into the cooperative behavior that solves the problem. A similar approach is reported in [22], where GP individuals contain partitioning instructions used for the creation of squads.

[9] These three papers are actually concurrent learning and hybrid team learning papers, not heterogeneous team learning work. However their arguments are germaine to the issue of heterogeneous specialists.

3 Concurrent Learning

The most common alternative to team learning in cooperative multi-agent systems is *concurrent learning*, where each agent on the team independently learns how to improve its performance and the performance of the team as a whole[10]. Whereas team learning decentralizes the agents while they are acting in the environment, concurrent learning additionally decentralizes the agents' learning procedures. Some research has presented domains where concurrent learning outperforms both homogeneous and heterogeneous team learning [40, 108, 109]. However, other investigations suggest that team learning may be preferable in certain situations [146]. Jansen and Wiegand [113] examine when the one method outperforms the other and vice versa.

There are many concurrent learning papers in reinforcement learning, but the notion is of special interest to evolutionary computation because of its close relationship with the *coevolution* with its multiple learners and independent interacting agents.

The primary advantage of concurrent learning is that it projects the large joint team search space onto 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 reductions in search space and in computational complexity. A second, 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, team learning experiments need only be concerned with a single learner, simplifying the design of the learning process itself.

The central challenge for concurrent 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, which in turn can ruin other agents' learned behaviors by making obsolete the assumptions on which they are based. 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-adaptation inherent in concurrent learning presents 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 in multi-agent environments with little or no communication.

Such co-adaptation creates many problems. One such problem is keeping the agents focused on the performance of the team, rather then on their individual performance. This issue may be cast in game theoretic terms: a multi-agent learning system can have multiple Nash equilibria, and once the set of agents reaches one of the equilibria, no single agent has a 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. Such equilibria, unfortunately, can also be suboptimal team behaviors.

Another problem: when centralized performance measures are used, how can the learning system(s) assess the contribution of each agent to the joint measure? This credit assignment problem may sometimes be very difficult to correctly solve or approximate. In some cases, equal shares of reward 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 solely 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.

Concurrent 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 concurrent learning (for example, Balch [10, 12] argues that the more local a reinforcement signal, the more homogeneous the final team). Further, relatively little concurrent learning work has been done in the area of communication. Thus we have broken the concurrent learning literature down into the following three groups. First: papers which analyze the *dynamics of learning*. The interest stems from the desire to better understand the individual learning processes and their influences on one another. Second: papers dealing with the impact of *locality of reinforcement* on the learned behaviors. The locality of reinforcement, 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 which deal with *modeling other agents* in order to improve the interaction with them.

[10] There are other degrees of granularity of course: the team may be divided into "squads", each with its own learner, for example. However, we are not aware of any concurrent learning literature which assigns learners to anything but individual agents.

3.1 The Dynamics of Learning

Many studies in concurrent learning have investigated the problem from a game-theoretic perspective, examining the game-theoretic dynamics of the learning algorithms. A important concept for such investigations is that of a Nash equilibrium, which is a joint strategy (one strategy for each agent) such that no single agent has any rational incentive (in terms of better payoff) to change its strategy away from the equilibrium. Many of the investigations of learning algorithms are concerned with proving that they converge to Nash equilibrium points. However, especially in approaches specific to collaborative multi-agent systems, an additional question regards whether the agents actually found a globally-optimal collaboration, or only a suboptimal Nash equilibrium point.

Repeated games A repeated game consists of a series of interactions among two or more agents. After each interaction, each agent may receive some reward (or punishment). Reinforcements for interactions are independent of any previous interactions. When all agents receive the same reinforcement each time (this may be thought of as a "team reinforcement"), the domain is called "fully cooperative". Through repeated interactions, each agent adjusts its strategy in order to maximize its received reward. Claus and Boutilier [47] propose two benchmark games (*climb* and *penalty*) and show 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 multi-agent learning algorithms with guarantees on convergence to optimal results. Boutilier [23], Chalkiadakis and Boutilier [46] suggest a Bayesian learning method for approximating the correct move each agent should make given a history of past interactions with the other agents, even in cases when the agent does not know the actions of the other agents that led to its reward.

Lauer and Riedmiller [125] suggest updating an agent's policy (Q-values) by estimating the likely best cooperation possible for the agent's action, and prove that this will converge to the optimum in deterministic environments. Kapetanakis and Kudenko [120, 119] 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. [166] point out that standard coevolutionary approaches, when applied to the same cooperative problem domains, are sometimes driven by *balance* considerations rather than performance. That is, agents tend to co-adapt to one another (including very poor collaborators) rather than trying to optimize their performance when in ideal conditions. Similar to Lauer, Panait *et al* show that evaluating an agent in the context with agents chosen as estimates of its best possible collaborators yields significantly improved results over the standard coevolutionary methods.

Littman and Stone [130] propose an algorithm for finding Nash equilibria points in two-player games. In a first step of their method, a feasible and enforceable reward combination for the two agents is identified. The next step consists of the construction of strategies that use punishments to avoid deviance from the specific rewards. The algorithm is polynomial in the number of actions for the two agents, but it is not guaranteed to find an optimal equilibrium. In the context of purely cooperative repeated games, the stochastic sampling algorithm proposed by Brafman and Tennenholtz [31] is the only one to our knowledge with guaranteed convergence to optimal Nash equilibria. The algorithm is polynomial in the number of actions of the agents, but it assumes *a priori* coordination of the agents' learning processes: the agents agree to a joint exploration phase of some length, then agree to a joint exploitation phase (agent settles on the behavior that yielded maximum reward).

In [160], two agents with two actions each participate in a repeated game with two Nash-equilibria points. However, each agent receives a different reinforcement (it's a non-symmetric game). In one of the Nash-equilibrium points, one agent receives more reward that the other agent, while in the other Nash-equilibrium point the situation is reversed. Thus any one given coordination strategy is unfair to one agent or to the other. Nowe et al. propose a "homo-equalis" reinforcement approach, where communication is used to alternate between the two unfair optimal points in order to fairly share the rewards received by the two agents.

Stochastic games Stochastic games are an extension of repeated games where at any point in time the game is in some *state*. The game transitions to a new state based on a stochastic function of the old state and the interactions among the agents in the old state. Reinforcements are based both on the interactions of the agents and the current state value. With a single state, a stochastic game reduces to a repeated game; with a single agent, a stochastic game reduces to a Markov decision process. Boutilier [24] describes *multi-agent Markov decision processes* (a particular subclass of stochastic games where all agents receive identical reinforcements) and suggests using decompositions via imposed conventions to simplify the coordination of individual learning processes. Bowling and Veloso [27] examine a number of game theory and reinforcement learning ap-

proaches to stochastic games, and describe the differences in assumptions they make. Hu and Wellman [103] (revised and extended by Bowling [25]) introduce a reinforcement learning algorithm for solving general-sum[11] stochastic games. Their learning algorithm assumes that the other agents are also learning using Q-learning; therefore, the agents do not learn just a single table of Q-values, but rather one such table for each agent. This extra information is used later to approximate the actions of the other agents. Nagayuki et al. [157] present an alternative approach where agents approximate the policies, rather than the tables of Q-values, of the other agents. Suematsu and Hayashi [227] point out that the algorithm of Hu and Wellman does not adapt to the other agent's policy, but only tries to converge to Nash equilibria. In the case when the other agent has a fixed policy, reaching a Nash equilibrium may be impossible. They then introduce the EXORL algorithm which learns optimal response policies[12] to both adaptive and fixed policies of the other agent. To our knowledge, Wang and Sandholm [241] present the only algorithm (*Optimal Adaptive Learning*) guaranteed to converge to optimal Nash equilibria in stochastic games. Their algorithm creates *virtual games* for each state in order to eliminate all strict suboptimal Nash equilibria.

Bowling and Veloso [29] describe two desirable properties for learning agents, namely rationality (the agent should converge optimally when the other agents have converged to stationary strategies) and convergence (under specified conditions, all agents should converge to stationary strategies). They then present a learning algorithm that exhibits these two properties. Bowling and Veloso [28] investigate the existence of equilibria points for agents with limitations which may prevent them from reaching optimality. Bowling and Veloso [30] introduce the WoLF algorithm (Win or Learn Fast), which varies the learning rate from small/cautious values when winning to large/aggressive values when losing to the other agents.

Peshkin et al. [173] investigate a distributed reinforcement learning approach to partially observable stochastic games with identical reinforcement to each agent, and show that their algorithm converges to local optima. Additionally, they show that there are local optima that are not necessarily Nash equilibria.

Other Analysis Coevolution may be cast into an evolutionary game theoretic framework. 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 [256, 258, 259, 260, 257]. Bull [38, 39] examines different learning parameters in coevolutionary algorithms. In [79, 182], a coevolutionary algorithm is augmented with a "hall of fame" repository consisting of the N best teams discovered so far. Individuals are evaluated by pairing them up with teammates chosen from this hall of fame; this approach outperforms ordinary cooperative coevolution methods.

Zhao and Schmidhuber [272], Schmidhuber [198], Schmidhuber and Zhao [199] examine the rates of rewards received for a given policy. If new policies yield lower reward rates over time, they are discarded and the learning algorithm proceeds with earlier-attempted policies. They call this the "success-story algorithm".

Last, Vidal and Durfee [238, 239] describe a framework to model and predict the behavior of multi-agent systems with learning agents. The system uses parameters such as rate of behavior change, learning and retention rates, and the influence agents have on one another, and then it charts the error in the agents' decision function. The authors show good predictions when analyzing learning algorithms introduced by other researchers.

Competition Though this survey focuses on cooperative multi-agent learning, we would be remiss if we failed to note some significant work on the game-theoretic dynamics of *competitive* multi-agent learning that may shed light on cooperative issues. Much of this work has been done in the realm of competitive coevolution (where N learners (subpopulations) compete, or where a single learner (population) competes against itself).

The goal of competitive coevolution is to establish an "arms race" among the agents that is beneficial to all of them. But unforeseen circumstances may prevent this from happening. One such problem arises from the subjective nature (i.e., in the context of other agents) of quality assessment; even if a given agent improves in some absolute sense, its performance metric relative to other agents may not increase, or may even *decrease*, depending on the gains made by its competitors. Many learning methods are highly sensitive to this performance metric, and so it can affect not only our external perception of agent improvement, but in fact the actual final result of the learning process. In competitive coevolution, this issue is known as the *Red-Queen effect* [48, 242, 71]. The name makes reference to Lewis Carroll's story "Through the Looking Glass", where the Red Queen is running in a moving landscape, and so appears not to be moving at all.

Another problem, *loss of gradient*, occurs when one learner comes to dominate the other learners in the sys-

[11] i.e., non-constant-sum: the reinforcements do not have to sum to the same constant value each time.

[12] At a Nash equilibrium point, all agents play optimal responses to the other agents' policies.

tem. In this situation, no learners receive sufficient feedback to improve — no matter what the dominant learner does, it always wins, and no matter what the subjugated learners do, they always lose [242, 176]. Last, competition can result in *cyclic behaviors*, where agents circle about one another due to non-transitive relationships between agent interactions. For example, in the rock-scissors-paper game, Agent A picks Rock and Agent B picks Paper. Agent A then adapts to Scissors, causing Agent B to adapt to Rock. Then Agent A adapts to Paper, Agent B adapts to Scissors, and Agent A adapts *again* to Rock. If a multi-agent environment is fraught with such non-transitive cycles, competing agents may be caught in them, rather than building the desired arms race [242, 71, 48, 191]

3.2 Credit Assignment and the Locality of Reinforcement

One facet of the credit assignment problem is the degree to which reinforcement should be local to individual agents. Consider a gold-mining problem, where reinforcement 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 loads 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 reinforcement methods tend to result in faster learning rates [10, 12]. But Balch shows that whether local or global reinforcement mechanisms lead to better performance tends to be problem-domain dependent. Balch [10, 12] compares the use of global (R_{global}) and local (R_{local}) reinforcement policies in foraging and simulated soccer tasks; 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 reinforcements lead to better performance in a foraging task [12], but better results are obtained when using global reinforcements in a simulated soccer domain [10]: teams of agents trained using R_{global} outscore a control team by an average of 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 reinforcement affects the heteorgeneity (diversity) of the learned behaviors. In both the foraging and simulated soccer domains, using local reinforcements results in homogeneous behaviors, while global reinforcements lead to heterogeneous teams. This and other work suggests that the degree of locality for the reinforcement should depend on other characteristics such as domain difficulty or number of skills required to solve the task [183, 11, 22, 180].

Wolpert and Tumer [263], Tumer et al. [235] propose a new method for assigning utility to individual agents in such a way that agents learning to maximize individual reward also have a strong interest in maximizing the global team reward. The new utility function is termed the *Wonderful Life Utility* (WLU) and approximates the utility of an agent by the inverse of the utility of the entire set of agents when the specific agent is not present. There results show that local rewards lead to worse performance than even selecting random strategies in some domains. This is explained by the agents' competition for immediate local rewards. Additionally, increasing the number of agents also increases their competition for immediate rewards, resulting in very poor scalability. Global rewards lead to slightly better performance than local rewards do, but WLU is better still, and also scales well with large numbers of agents. The authors also show that local rewards may lead to convergence to Nash equilibria that do not coincide with globally optimal team performance; instead, WLU will usually produce Nash equilibria that coincide with optimal team behavior.

Mataric [138] argues that agents' separate learning processes can be improved by combining individual local reinforcement information (based on some measure of progress towards one's own goals) with other types of *social reinforcement*. One such type of information is obtained by observing other agents and imitating their behaviors; this may improve the overall team behavior by reproducing rare behaviors. This leads to an *observational reinforcement* which agents receive whenever repeating other agents' behaviors. *Vicarious reinforcement*, a third component of the individual agent reinforcement, consists of small agent reinforcements based on those received by other agents; its purpose is to spread individual reinforcement to other agents and to balance local and global reinforcement information. Mataric shows that a weighted combination of these three components gives better results in a foraging application.

Tangamchit et al. [232] suggest that discounting rewards (considering immediate rewards to be more important than future ones) leads to robots learning greedy actions designed for immediate reinforcement, hampering better collaborations. They compare local and global reinforcements in combination with average (Monte-Carlo) or discounted (Q-Learning) reinforcement in a foraging domain. The results show that average reinforcement combined with global reinforcement leads to much better results than discounting.

3.3 Teammate Modeling

A final area of research is 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 [237] 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 general, an N-level agent models its teammates as (N-1)-level agents.

Mundhe and Sen [155] 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 [210, 209], where a couple of robots successfully learn to cooperatively push a box without either even being aware of the other's presence. Mukherjee and Sen [152], Banerjee et al. [13] present experiments in which 1-level agents model each others' action probability distribution; this produces a form of mutual trust.

When dealing with larger numbers of teammates, a simpler modeling approach is to presume that the entire team consists of agents identical to the modeling agent. Haynes and Sen [93, 91], Haynes et al. [89] use this approach complemented with a case-based learning mechanism for dealing with exceptions from this assumed behavior.

Hu and Wellman [102], Wellman and Hu [251] suggest that, when learning models of other agents in a multi-agent learning scenario, the resulting behaviors are highly sensitive to the agents' initial beliefs,. Depending on these initial beliefs, the final performance may be better or even *worse* 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 [104]: the authors suggest the best policy for creating learning agents is to *minimize* the assumptions about the other agents' policies.

Suryadi and Gmytrasiewicz [228] 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 usually be 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.

One aspect of modeling is attempting to discover if the other agents in the environment are cooperating with you, competing with you, or in some other relationship with you. Sekaran and Sen [203], Sen and Sekaran [207] investigate the application of reciprocity concepts to multi-agent domains where not enough information on the intentions of other agents is known *a priori*. They apply a stochastic decision-making algorithm that encourages reciprocity among agents while avoiding being taken advantage of by unfriendly agents. The authors show that the agents that prefer not to cooperate end up with worse performance. Nowak and Sigmund [159] mention two types of reciprocity: direct (agent A helps another agent B and expects B to help him back sometime in the future) and indirect (an agent helps another in return for future help from other agents). Such reciprocity improves an agent's "reputation". They argue that a necessary condition for cooperation is that the "degree of acquaintanceship" (the probability that an agent knows another agent's reputation) should exceed the ratio of cost of altruist help relative to the benefit to the recipient. Similar analysis can be done without the need to model the reputation *per se*: Ito [112] pitted game-players against others in the population, where each game player had access to a history of his opponent's previous actions against other players.

We conclude this section with work in agent modeling under communication. Ohko et al. [161] use a communication protocol for agents to subcontract subtasks to other agents. In their approach, each agent tries to decompose tasks into simpler subtasks and broadcasts announcements about the subtasks to all other agents in order to find "contractors" who can solve them more easily. 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 [37, 36]: 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.

4 Learning and Problem Decomposition

The state space of a large, joint multi-agent task can be overwhelming. An obvious way to tackle this is to use domain knowledge to simplify the state space, often by providing a smaller set of more "powerful" actions customized for the problem domain. For example, Mataric [143, 140] applies Q learning to select from hand-coded reactive behaviors such as *avoid*, *head-home*, *search* or *disperse* for robot foraging tasks.

An alternative, which we discuss through the remainder of this section, is to reduce complexity by heuristically decomposing the problem, and hence the *learned behavior*, into separate, simpler subtasks for the agents to learn. 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 is 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.[13] Each squad may be endowed with a different set of 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 [133, 134]. The number of such squads can also be learned, giving the team some freedom in problem decomposition [87, 22].

Layered Learning In layered learning, each individual behavior is decomposed into a hierarchy of sub-behaviors, each sub-behavior using the ones beneath it. Some of the behaviors may be hardcoded, while others are modified by learning processes. This method was proposed by Stone [223, 222] and successfully applied to robotic soccer. Further applications of the technique are reported by Gustafson [85], Gustafson and Hsu [86], Hsu and Gustafson [101]: the authors present an 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.

Hierarchical Learning Makar et al. [137] suggest a different approach to simplifying the inter-agent coordination task. Each agent learns three interrelated *skills*: solving subtasks, ordering subtasks, and coordinating with other agents. This hierarchical approach allows agents to only coordinate based on information at higher levels in the hierarchy: for example, instead of learning what an agent should do for each combination of elementary actions the other teammates can perform, the agents decide their strategies in the context of being aware of the current higher-level subtasks of their teammates. This implies that agents do not need to know details about what other agents are currently doing, but only abstract information about the subtasks they are trying to solve. This approach dramatically reduces the communication requirements and eases learning.

Behavior Decomposition Zhang and Cho [271] suggest that the learning task may be simplified by decomposing the desired behavior into simpler components that can be evaluated and learned separately; they term this approach *fitness switching*. The algorithm assumes a homogeneous team of agents; the behavior of each agent is separated into several simpler sub-behaviors. The performance of a team is assessed based on a combination (sum in their suggested algorithm) of performances to accomplishing the simpler sub-behaviors. The results suggest that fitness switching is a better alternative to simple genetic programming in domains where the task can be decomposed. Behaviors may also be decomposed vertically (into ordered sequences of sub-behaviors rather than sets of them). To this end, Balch [12] uses a shaped reinforcement reward function (R_{shaped}) (earlier suggested by Mataric [139]) which depends on the number of partial steps fulfilled for accomplishing the task. Balch shows that using R_{shaped} leads to similar results to using a local reinforcement function R_{local}, but in a significantly shorter time. The results are similar to those of best hand-coded solutions, and also better than those obtained when using a global reinforcement function R_{global}.

Team Behavior Decomposition Guestrin et al. [84] note that in many domains the actions of some agents may be independent. Taking advantage of this, they suggest partially decomposing the joint Q-values of agents based on a *coordination graph* that heuristically spells

[13] Indeed one might argue that a homogeneous multi-agent 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.

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.

5 Learning and Communication

For some problems communication is a necessity; for others, communication may nonetheless 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 [225] argue that unrestricted communication reduces a multi-agent system to something isomorphic to a single-agent system. They do this by noting that without any restriction, the agents can send complete external state information to a "central agent", and to execute its commands in lock-step, in essence acting as effectors for the central agent. A central agent is not even necessary: as long as agents can receive all the information they need to known 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 [66], this increase in search space can hamper learning an optimal behavior by more than communication itself may 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 [135] assume that predators can sense each other's position no matter what distance separates them, and Berenji and Vengerov [20] use a blackboard communication scheme to allow agents to know of other agents' locations.

5.1 Direct Communication

Many agent communication methods employ, or assume, an external communication method by which agents may share information with one another. The method may be constrained in terms of bandwidth, locality, agent class, etc. Examples of direct communication include shared blackboards, signaling, and message-passing. The literature has examined both hard-coded communication methods and learned communication methods, and their effects on cooperative learning overall.

Tan [231] suggests that cooperating learners can use communication in a variety of ways in order to improve team performance. For example, agents can inform others of their current state by sharing immediate sensor information. Another approach is for agents to share information about past experiences in the form of episodes (sequences of $\langle state, action, reward \rangle$ previously encountered by the agent) that others may not have experienced yet. Yet another alternative is for agents to share knowledge related to their current policies (for example, in the form of $\langle state, action, utility \rangle$ for cooperating reinforcement learning agents). Tan suggests that cooperating agents that share information about current sensor readings, past experiences, and current learned policies, significantly outperform independent learning agents. Of course, this increased performance is obtained at the expense of increased communication usage necessary for the exchange of information.

Some research, particularly in reinforcement learning, has simply 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. For example, Berenji and Vengerov [19] investigate a cooperative learning setting where multiple agents employ communication to use and update the same policy. Berenji and Vengerov suggest that the simultaneous update of a central policy reduces early tendencies for convergence to suboptimal behaviors. We argue that this is an implicit hard-coded communication procedure: the learners are teaching each other learned information.

Much of the remaining research provides the agents with a communication channel but does not hard-code its purpose. In a simple situation, agents' vocabulary may consist of a single signal detectable by other agents [240]. In other work (for example [269]) mobile robots

in a cooperative movement task are endowed with a fixed but undefined communication vocabulary. The robots learn to associate meanings with words in various trials. When circumstances change, the robots learn to adjust their communication language as appropriate. A similar approach is reported in [116], where a genetic algorithm is used for language learning in a predator-prey domain. The authors use a blackboard communication scheme and show that increasing the language size improves 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 multi-agent problems. Steels [216] reports the emergence of a spontaneous coherent lexicon that may adapt to cope with new meanings during the lifetime of the agents. Steels and Kaplan [221] continue this investigation to show that agents are able to create general words through collective agreement on their meanings and coverage. Similar approaches to evolving communication languages are presented in [215, 217, 218, 195, 57, 219, 220, 42].

Many such methods provide an incentive to the agents to communicate (perhaps by sharing the resulting reward). However, Ackley and Littman [1] show that reward incentives are not necessary for the agents to communicate: instead, agents learn to communicate even when the agents receive no benefit from this action.

Many agent communication methods employ, or assume, an external communication method by which agents may share information with one another. The method may be constrained in terms of bandwidth, locality, agent class, etc. Examples of direct communication include shared blackboards, signaling, and message-passing. The literature has examined both hard-coded communication methods and learned communication methods, and their effects on cooperative learning overall.

5.2 Indirect Communication

We define indirect communication methods as those which involve the *implicit* transfer of information from agent to agent through modification of the world environment. Examples of indirect communication include: leaving footsteps in snow, leaving a trail of bread crumbs in order to find one's way back home, and providing hints through the placement of objects in the environment (perhaps including the agent's body itself).

Much of the indirect communication literature has drawn inspiration from social insects' use of pheromones to mark trails or to recruit other agents for tasks [100]. Pheromones are chemical compounds whose presence and concentration can be sensed by fellow insects [21], and like many other media for indirect communication, pheromones can last a long time in the environment, though they may diffuse or evaporate. In some sense, pheromone deposits may be viewed as a large blackboard or state-utility table shared by all the agents; but they are different in that pheromones can only be detected locally.

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 [126, 149, 151, 148, 150]. Evolutionary computation techniques have also been applied to learn exploration/exploitation strategies using pheromones deposited by hardcoded mechanisms. For example, Sauter et al. [197, 196] 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 [255].

Another research direction is concerned with studying 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 [50, 49]. AntFarm ants use a single pheromone to mark trails toward food sources, but use a compass to point themselves along the shortest path back to the nest. Panait and Luke [164] present a related algorithm that exhibits good performance at various foraging tasks in the presence of obstacles. 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 guide themselves to the food source and back to the nest. The authors note a hill-climbing effect that leads to straight (and locally optimal) paths. In an extension of their work, Panait and Luke [165] successfully apply evolutionary computation to learn a pheromone-based foraging task. 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 shows that behaviors learned for more complex domains exhibit good performance in simpler problems as well.

Werger and Mataric [253] present another approach to indirect communication among agents: using the agents' body positions themselves to convey meaning. The authors present a foraging application where a number of robots need to collect items and deposit them at a prespecified location. The robots use their bodies to mark the path for other robots to follow. In some sense, the robots' bodies are acting essentially as a pheromone trail

for the other robots.

Quinn [184] argues that this is a form of communication by citing Wilson's statement that communication "is neither the signal by itself, nor the response, [but] instead the relationship between the two" [262]. This definition suggests that a shared dedicated channel is not necessary for communication. Quinn [184] investigates the 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".

6 Two Challenges: Scalability and Adaptive Dynamics

Multi-agent 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 multi-agent learning more broadly successful as a technique. Both of these challenges arise from the *multi* in multi-agent 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.

6.1 Scalability

Scalability is a problem for many learning techniques, but especially so in multi-agent 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 multi-agent 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.

6.2 Adaptive Dynamics

Multi-agent systems are typically dynamic environments, with multiple learning agents vying for resources and tasks. This dynamicism presents a unique challenge not normally found in single-agent learning: as the agents learn, their adaptation to one another changes the world scenario. How do agents learn in an environment where the goalposts are constantly and adaptively 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 multi-agent systems may be thought of in game-theoretic terms, suggesting a convergence to stable equillibria; as opposed to actual optima. Researchers have argued that an important part of cooperative multi-agent learning is figuring out how to adjust the learning methods or the problem domain such that the agents are likely to converge to desired Nash equilibria, especially ones which are near-optimal [241, 31, 260]. For example, cooperative coevolution has revealed that the multi-agent learning process is sometimes guided by "balance" considerations, rather than performance improvements. The adaptation of single-agent learning algorithms to the multi-agent 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. [166] show 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 multi-agent teams. That is, the closer one can get to assessing individuals with their optimal collaborators, the more likely one is to optimizing the system in an evolutionary computation sense.

7 Problem Domains and Applications

Despite the relative young age of the field, the multi-agent systems area contains a very large number of problem domains. The list in this survey is far from complete, but it contains many of the common problems. The problem domains are divided in three classes: embodied agents, game-theoretic environments, and applications to real-world problems.

7.1 Embodied Agents

The cost of robots has 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 multi-agent 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.

Predator-Prey Pursuit This is one of the most common environments in multi-agent learning research, and 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. The earliest work using this domain is [18], but many there are many variations of the problem [63].

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, and drop location. The efficiency of an approach may be defined by how quickly it completes the foraging task [142, 54], or by the number of items collected in a fixed amount of time. Ostergaard et al. [162] provide an extended survey of previous work in this domain. A variation of the problem allows agents to communicate by depositing pheromones in the environment to mark trails connecting rich pick-up and drop-off areas [21, 164, 165]. In a related problem domain, *clustering*, the agents have no pre-specified drop locations; rather, they only need to pile all the items together [17], or sort them by class into separate piles [62].

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 pushing them. Sometimes a robot is capable of pushing one box by itself (see for example the single-agent experiments reported in [136, 35]). A more complicated version requires two or more agents to cooperate in order to move a single box in simulation [271], or on real robots [141, 140]. Buffet et al. [34], Dutech et al. [69] present reinforcement learning approaches for a similar problem domain where agents receive rewards only when they merge together pucks of different colors.

Soccer The game of soccer, both in simulation and with real robots, is one of the most widely used domains for research in multi-agent systems [123]. 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. Other performance metrics have included length of time in control of the ball, successful interceptions, and average location of the ball. Successful applications of learning techniques to this domain are reported in [134, 222, 11, 225, 189]. The strong interest in this domain has led to several annual "world cup" robot soccer championships. The most well-known such competition, RoboCup, has different leagues divided by continent and by type of robot or simulation environment. RoboCup's goal is "by the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team" (see *www.robocup.org*).

Keep-Away Soccer Gustafson and Hsu [86] 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 [224].

Cooperative Navigation The task, as described in [11], 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 behaviors, and the performance is assessed based on the maximum time necessary for the agents to accomplish the task. The questions investigated by Balch include benefits from formation participation and the impact of diversity on performance. The Opera Problem, discussed in [51], represents another challenge problem in cooperative navigation: a large number of agents located in a bounded space need to coordinate in order to quickly leave it through a small fixed exit area. An initial homogeneous team learning approach to this problem is presented in [190].

Cooperative Target Observation Introduced in [167], 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 [70, 168, 170]. A related problem domain is described in [268, 16, 15]: a team of agents needs to perform a surveillance task in a region of space. The agents coordinate to avoid collisions, and their reward is further increased when discovering and observing areas of high interest. Because of constraints placed on flying unmanned vehicles, the agents need to maintain a minimal speed that would keep them in the air.

Herding Schultz et al. [201] introduce a new domain where a robot must herd another robot into a designated area. The second robot (the sheep) will avoid obstacles and the "shepherd" robot, but otherwise may either move randomly or try to avoid the herding area. The shepherd moves close to the sheep robot to encourage the sheep to move in the desired direction. Potter et al. [180] try a multi-shepherd version of the domain, where many faster and "stubborn" sheep require coordination from the several herding agents. Additionally, predators may exist in the environment and try to kill the sheep while avoiding the shepherds, thus complicating the shepherds' task. Another application of learning techniques to a herding domain is presented in [254].

7.2 Game-Theoretic Environments

As discussed in Section 6.2, many multi-agent 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 multi-agent 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 each agent to learn from the other's behavior and to adapt its actions appropriately. The Iterated Prisoner's Dilemma is considered a *cooperative* game because the better pairs of agents tend to cooperate with one another [7, 8]. A three-person coordination game inspired by the Iterated Prisoner's Dilemma is presented in [2].

Coordination Games Various repeated games described in terms of joint reward matrices have been previously introduced in the literature to highlight specific issues associated with multi-agent learning. For example, Claus and Boutilier [47] introduce two simple 3x3 matrix games: a coordination game with two optima and high penalties for miscoordination; and a second game with two Nash-equilibrium points, one of them corresponding to a suboptimal collaboration. These games are later used in [125, 120, 166] to investigate multi-agent reinforcement learning and evolutionary computation approaches.

Social Dilemmas These problems concern the individual decisions of several agents, all of which receive a joint reward [75]. 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 these two problems, accompanied by a coevolutionary approach to learning solutions to them, can be found in [156]. 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 [6, 265].

7.3 Real-World Applications

This section describes a set of problems drawn from real-world domains previously used 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 primarily 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 [128].

Air Traffic Control For security purposes, the airspace 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 multi-agent approach for this domain is reported in [214].

Network Management and Routing This domain consists of a large, distributed network. Agents are deployed to distributively and cooperatively control and manage the network, handle failures, and balance its load [243]. Littman and Boyan [129], Subramanian et al. [226], Wolpert et al. [264] introduce various learning approaches to network routing.

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 [236]. Schneider et al. [200] present a reinforcement learning approach to managing a power grid.

Distributed Medical Care This domain applies AI to assist clinical staff in making diagnoses, decide on therapy and tests, determine prescriptions, and perform other health care tasks [105]. The problem is particularly suited for multi-agent 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 [266]. Brauer and Weiß [32] present a learning approach for such a domain.

Hierarchical Multi-Agent Systems Problems Some multi-agent 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 may 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 multi-agent approach to this domain is reported in [72]. 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 [154]. A related "Factory Floor" problem is investigated in [172], 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 work in this area include hard-coded and learned collective behaviors such as flocking [188, 186, 212, 213], learning of dispersion and aggregation [270], learning social behaviors in virtual worlds [80, 81], and the modeling of the formation of early countries [45].

Other multi-agent systems domains investigated include particle accelerator control [115], intelligent document retrieval [153], spacecraft control [202], and concurrent engineering [53]. Further applications of Distributed AI in industry are reported in [171].

8 Resources

Machine learning in multi-agent systems has seen a large increase in the number of research papers published in the last few years. General material on the topic can be found in [245, 111, 250, 204, 246, 107, 247, 206, 248]. Other surveys of existing work in multi-agent learning and related domains include [61, 68, 33, 67, 244, 171, 43, 205, 132, 249, 60, 127, 225, 169]. There are also several PhD theses investigating different aspects of the multi-agent learning field: [76, 142, 64, 52, 177, 44, 82, 158, 11, 223, 26, 257] Good pointers for news, conferences and journals, courses, people, companies, and laboratories may be found on the web sites http://www.multiagent.com and http://agents.umbc.edu.

9 Conclusions

Cooperative multi-agent learning is a relatively new area, fraught with complex and rich dynamics and with scal-

ability problems, but which also holds the promise of widespread applicability. This paper surveyed the multi-agent learning area, presenting a novel categorization of previous work, accompanied by discussions on key aspects of interest. We argued that learning in multi-agent systems can be done at a "team" level, where a single learner improves the behavior of the entire team, and at a "teammate" level, where individual agents conduct their own learning processes. The existence of, or lack of, multiple learners is of enough impact on the method that the team learning and concurrent learning literature break down along rather different lines.

We chose to divide the team learning literature into approaches to learn behaviors for homogeneous, heterogeneous, and hybrid homogeneous/heterogeneous teams. We then surveyed concurrent learning, dividing the literature into papers investigating the optimality of various learning approaches, the impact of locality of reinforcement information, cooperation or competition among agents, and modeling other agents in the domain. Later sections dealt with the issues of problem decomposition and communication. We then finished with some challenge issues, problem domains, and resources for the multi-agent learning community.

10 Acknowledgements

The authors would like to thank Ken De Jong, Paul Wiegand, Gabriel Balan, Jeff Bassett, Mitch Potter, Valentin Prudius, Jayshree Sarma, Marcel Barbulescu, Adrian Grajdeanu, Elena Popovici, Keith Sullivan, Zbigniew Skolicki and Joerg Denzinger for help in preparing this survey.

References

[1] D. H. Ackley and M. L. Littman. Altruism in the evolution of communication. In *Artificial Life IV: Proceedings of the International Workshop on the Synthesis and Simulation of Living Systems, third edition*. MIT Press, 1994.

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

[3] D. Andre, F. Bennett III, and J. Koza. 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, 1996.

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

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

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

[7] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

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

[9] T. Bäck. *Evolutionary Algorithms in Theory and Practice: Evolutionary Straegies, Evolutionary Programming, and Genetic Algorithms*. Oxford Press, 1996.

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

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

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

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

[14] A. G. Barto, R. S. Sutton, and C. Watkins. Learning and sequential decision making. In M. Gabriel and J. Moore, editors, *Learning and computational neuroscience : foundations of adaptive networks*. M.I.T. Press, Cambridge, Mass, 1990.

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

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

[17] R. Beckers, O. E. Holland, and J.-L. Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. In *Artificial Life IV: Proceedings of the International Workshop on the Synthesis and Simulation of Living Systems , third edition*. MIT Press, 1994.

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

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

[20] H. Berenji and D. Vengerov. 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, 2000.

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

[22] J. C. Bongard. The legion system: A novel approach to evolving heterogeneity for collective problem solving. In R. Poli, W. Banzhaf, W. B. Langdon, J. F. Miller, P. Nordin, and T. C. Fogarty, editors, *Genetic Programming: Proceedings of EuroGP-2000*, volume 1802, pages 16–28, Edinburgh, 15-16 2000. Springer-Verlag. ISBN 3-540-67339-3.

[23] C. Boutilier. Learning conventions in multiagent stochastic domains using likelihood estimates. In *Uncertainty in Artificial Intelligence*, pages 106–114, 1996.

[24] C. Boutilier. 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, 1996.

[25] M. Bowling. 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, 2000.

[26] M. Bowling. *Multiagent learning in the presence of agents with limitations*. PhD thesis, Computer Science Department, Carnegie Mellon University, 2003.

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

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

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

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

[31] R. Brafman and M. Tennenholtz. Efficient learning equilibrium. In *Advances in Neural Information Processing Systems (NIPS-2002)*, 2002.

[32] W. Brauer and G. Weiß. Multi-machine scheduling - a multi-agent learning approach. In *Proceedings of the Third International Conference on Multi-Agent Systems*, pages 42–48, 1998.

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

[34] O. Buffet, A. Dutech, and F. Charpillet. Incremental reinforcement learning for designing multi-agent systems. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 31–32, Montreal, Canada, 2001. ACM Press.

[35] O. Buffet, A. Dutech, and F. Charpillet. Learning to weigh basic behaviors in scalable agents. In

Proceedings of the 1st International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS'02), 2002.

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

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

[38] L. Bull. Evolutionary computing in multi-agent environments: Partners. In T. Back, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 370–377. Morgan Kaufmann, 1997.

[39] L. Bull. Evolutionary computing in multi-agent environments: Operators. In D. W. V W Porto, N Saravanan and A. E. Eiben, editors, *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, pages 43–52. Springer Verlag, 1998.

[40] L. Bull and T. C. Fogarty. Evolving cooperative communicating classifier systems. In A. V. Sebald and L. J. Fogel, editors, *Proceedings of the Fourth Annual Conference on Evolutionary Programming (EP94)*, pages 308–315, 1994.

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

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

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

[44] D. Carmel. *Model-based Learning of Interaction Strategies in Multi-agent systems*. PhD thesis, Technion - Israel Institute of Technology, 1997.

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

[46] G. Chalkiadakis and C. Boutilier. Coordination in multiagent reinforcement learning: A Bayesian approach. In *Proceedings of The Second International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS 2003)*. ACM, 2003. ISBN 1-58113-683-8.

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

[48] D. Cliff and G. F. Miller. 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, 1995.

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

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

[51] V. Crespi, G. Cybenko, M. Santini, and D. Rus. Decentralized control for coordinated flow of multi-agent systems. Technical Report TR2002-414, Dartmouth College, Computer Science, Hanover, NH, January 2002.

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

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

[54] T. S. Dahl, M. J. Mataric, and G. S. Sukhatme. Adaptive spatio-temporal organization in groups of robots. In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-02)*, 2002.

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

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

[57] B. de Boer. Generating vowel systems in a population of agents. In *Proceedings of the Fourth European Conference Artificial Life*. MIT Press, 1997.

[58] K. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Arbor, MI, 1975.

[59] K. De Jong. *Evolutionary Computation: A unified approach*. MIT Press, 2003.

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

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

[62] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Chretien. The dynamics of collective sorting: Robot-like ants and ant-like robots. In *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pages 356–363. MIT Press, 1991.

[63] J. Denzinger and M. Fuchs. Experiments in learning prototypical situations for variants of the pursuit game. In *Proceedings on the International Conference on Multi-Agent Systems (ICMAS-1996)*, pages 48–55, 1996.

[64] M. Dowell. *Learning in Multiagent Systems*. PhD thesis, University of South Carolina, 1995.

[65] G. Dudek, M. Jenkin, R. Milios, and D. Wilkes. A taxonomy for swarm robots. In *Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems*, 1993.

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

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

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

[69] A. Dutech, O. Buffet, and F. Charpillet. Multi-agent systems by incremental gradient reinforcement learning. In *Proceedings of Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 833–838, 2001.

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

[71] S. Ficici and J. Pollack. Challenges in coevolutionary learning: Arms–race dynamics, open–endedness, and mediocre stable states. In C. Adami *et al*, editor, *Proceedings of the Sixth International Conference on Artificial Life*, pages 238–247, Cambridge, MA, 1998. MIT Press.

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

[73] D. Fogel. *Blondie24: Playing at the Edge of Artificial Intelligence*. Morgan Kaufmann, 2001. ISBN 1-55860-783-8.

[74] L. Fogel. *Intelligence Through Simulated Evolution: Fourty Years of Evolutionary Programming*. Wiley Series on Intelligent Systems, 1999.

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

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

[77] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, Reading, MA, 1989.

[78] C. V. Goldman and J. S. Rosenschein. Mutually supervised learning in multiagent systems. In G. Weiß and S. Sen, editors, *Adaptation and Learning in Multi-Agent Systems*, pages 85–96. Springer-Verlag: Heidelberg, Germany, Berlin, 1996.

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

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

[81] S. Grand and D. Cligg. Creatures: Entertainment software agents with artificial life. *Autonomous Agents and Multi-Agent Systems*, 1(1): 39–57, 1998.

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

[83] J. J. Grefenstette. Lamarckian learning in multiagent environments. In R. Belew and L. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 303–310, San Mateo, CA, 1991. Morgan Kaufman.

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

[85] S. M. Gustafson. Layered learning in genetic programming for a co-operative robot soccer problem. Master's thesis, Kansas State University, Manhattan, KS, USA, 2000.

[86] S. M. Gustafson and W. H. Hsu. Layered learning in genetic programming for a co-operative robot soccer problem. In J. F. Miller, M. Tomassini, P. L. Lanzi, C. Ryan, A. G. B. Tettamanzi, and W. B. Langdon, editors, *Genetic Programming: Proceedings of EuroGP-2001*, volume 2038, pages 291–301, Lake Como, Italy, 18-20 2001. Springer-Verlag. ISBN 3-540-41899-7.

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

[88] I. Harvey, P. Husbands, D. Cliff, A. Thompson, and N. Jakobi. Evolutionary robotics : the Sussex approach. *Robotics and Autonomous Systems*, 1996.

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

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

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

[92] T. Haynes and S. Sen. Cooperation of the fittest. Technical Report UTULSA-MCS-96-09, The University of Tulsa, Apr. 12, 1996.

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

[94] T. Haynes and S. Sen. Crossover operators for evolving a team. In J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 162–167, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann.

[95] T. Haynes, S. Sen, D. Schoenefeld, and R. Wainwright. Evolving a team. In E. V. Siegel and J. R. Koza, editors, *Working Notes for the AAAI Symposium on Genetic Programming*, pages 23–30, MIT, Cambridge, MA, USA, 10–12 Nov. 1995. AAAI.

[96] T. Haynes, S. Sen, D. Schoenefeld, and R. Wainwright. Evolving multiagent coordination strategies with genetic programming. Technical Report UTULSA-MCS-95-04, The University of Tulsa, May 31, 1995.

[97] T. Haynes, R. Wainwright, S. Sen, and D. Schoenefeld. Strongly typed genetic programming in evolving cooperation strategies. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, pages 271–278, Pittsburgh, PA, USA, 15-19 July 1995. Morgan Kaufmann. ISBN 1-55860-370-0.

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

[99] J. Holland. *Adaptation in Natural and Artificial Systems*. The MIT Press, Cambridge, MA, 1975.

[100] B. Hölldobler and E. O. Wilson. *The Ants*. Harvard University Press, 1990.

[101] W. H. Hsu and S. M. Gustafson. Genetic programming and multi-agent layered learning by reinforcements. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska, editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 764–771, New York, 9-13 July 2002. Morgan Kaufmann Publishers. ISBN 1-55860-878-8.

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

[103] J. Hu and M. P. Wellman. 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, 1998.

[104] J. Hu and M. P. Wellman. Online learning about other agents in a dynamic multiagent system. In K. P. Sycara and M. Wooldridge, editors, *Proceedings of the Second International Conference on Autonomous Agents (Agents'98)*, pages 239–246, New York, 1998. ACM Press. ISBN 0-89791-983-1.

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

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

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

[108] H. Iba. Emergent cooperation for multiple agents using genetic programming. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature IV: Proceedings of the International Conference on Evolutionary Computation*, volume 1141 of *LNCS*, pages 32–41, Berlin, Germany, 22-26 Sept. 1996. Springer Verlag. ISBN 3-540-61723-X.

[109] H. Iba. Evolutionary learning of communicating agents. *Information Sciences*, 108, 1998.

[110] H. Iba. Evolving multiple agents by genetic programming. In L. Spector, W. Langdon, U.-M. O'Reilly, and P. Angeline, editors, *Advances in Genetic Programming 3*, pages 447–466. The MIT Press, Cambridge, MA, 1999.

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

[112] A. Ito. How do selfish agents learn to cooperate? In *Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, pages 185–192. MIT Press, 1997.

[113] T. Jansen and R. P. Wiegand. Exploring the explorative advantage of the cooperative coevolutionary (1+1) EA. In E. Cantu-Paz *et al*, editor, *Prooceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. Springer-Verlag, 2003.

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

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

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

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

[118] L. P. Kaelbling, M. L. Littman, and A. P. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[119] S. Kapetanakis and D. Kudenko. 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)*, 2002.

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

[121] G. Kendall and G. Whitwell. 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, 27-30 2001.

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

[123] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa. RoboCup: The robot world cup initiative. In W. L. Johnson and B. Hayes-Roth, editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 340–347, New York, 5–8, 1997. ACM Press. ISBN 0-89791-877-0.

[124] J. Koza. *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

[125] M. Lauer and M. Riedmiller. 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, 2000.

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

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

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

[129] M. Littman and J. Boyan. A distributed reinforcement learning scheme for network routing. Technical Report CS-93-165, Carnegie Mellon University, 1993.

[130] M. Littman and P. Stone. A polynomial-time Nash equilibrium algorithm for repeated games, 2003.

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

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

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

[134] S. Luke, C. Hohn, J. Farris, G. Jackson, and J. Hendler. 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, 1997.

[135] S. Luke and L. Spector. Evolving teamwork and coordination with genetic programming. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pages 150–156, Stanford University, CA, USA, 28–31 1996. MIT Press.

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

[137] R. Makar, S. Mahadevan, and M. Ghavamzadeh. Hierarchical multi-agent reinforcement learning.

In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 246–253, Montreal, Canada, 2001. ACM Press.

[138] M. Mataric. Learning to behave socially. In *Third International Conference on Simulation of Adaptive Behavior*, 1994.

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

[140] M. Mataric. Using communication to reduce locality in distributed multi-agent learning. *Joint Special Issue on Learning in Autonomous Robots, Machine Learning, 31(1-3), 141-167, and Autonomous Robots, 5(3-4), Jul/Aug 1998, 335-354*, 1998.

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

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

[143] M. J. Mataric. Reward functions for accelerated learning. In *International Conference on Machine Learning*, pages 181–189, 1994.

[144] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin, 3rd edition, 1996.

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

[146] T. Miconi. 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)*, 2003.

[147] M. Mitchell, J. Crutchfield, and R. Das. 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),*, 1996.

[148] N. Monekosso, P. Remagnino, and A. Szarowicz. An improved Q-learning algorithm using synthetic pheromones. In E. N. B. Dunin-Keplicz, 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, 2002.

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

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

[151] N. D. Monekosso, P. Remagnino, and A. Szarowicz. 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, 2001.

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

[153] U. Mukhopadjyay, L. Stephens, and M. Huhns. An intelligent system for document retrieval in distributed office environment. *Journal of the American Society for Information Science*, 37, 1986.

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

[155] M. Mundhe and S. Sen. Evaluating concurrent reinforcement learners. In *Proceedings of the International Conference on Multiagent System*, 2000.

[156] M. Mundhe and S. Sen. Evolving agent societies that avoid social dilemmas. In D. Whitley, D. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, and H.-G. Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pages 809–816, Las Vegas, Nevada, USA, 10-12 2000. Morgan Kaufmann. ISBN 1-55860-708-0.

[157] Y. Nagayuki, S. Ishii, and K. Doya. 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), 2000.

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

[159] M. Nowak and K. Sigmund. Evolution of indirect reciprocity by image scoring / the dynamics of indirect reciprocity. *Nature*, 393:573–577, 1998.

[160] A. Nowe, K. Verbeeck, and T. Lenaerts. Learning agents in a homo egualis society. Technical report, Computational Modeling Lab - VUB, March 2001.

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

[162] E. H. Ostergaard, G. S. Sukhatme, and M. J. Mataric. 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*,, 2001.

[163] L. Pagie and M. Mitchell. A comparison of evolutionary and coevolutionary search. In R. K. Belew and H. Juillè, editors, *Coevolution: Turning Adaptive Algorithms upon Themselves*, pages 20–25, San Francisco, California, USA, 7 2001.

[164] L. A. Panait and S. Luke. Ant foraging revisited. In *Proceedings of the Second International Workshop on the Mathematics and Algorithms of Social Insects*, Atlanta, Georgia, 2003.

[165] L. A. Panait and S. Luke. Evolving foraging behaviors. In *Proceedings of the Second International Workshop on the Mathematics and Algorithms of Social Insects*, Atlanta, Georgia, 2003.

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

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

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

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

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

[171] H. V. D. Parunak. Applications of distributed artificial intelligence in industry. In G. M. P. O'Hare and N. R. Jennings, editors, *Foundations of Distributed AI*. John Wiley & Sons, 1996.

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

[173] L. Peshkin, K.-E. Kim, N. Meuleau, and L. P. Kaelbling. Learning to cooperate via policy search. In *Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 307–314. Morgan Kaufmann, 2000.

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

[175] J. Pollack, A. Blair, and M. Land. Coevolution of a backgammon player. In *Artificial Life V*. MIT Press, 1997.

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

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

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

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

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

[181] M. A. Potter, K. A. De Jong, and J. J. Grefenstette. 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., 1995.

[182] N. Puppala, S. Sen, and M. Gordin. Shared memory based cooperative coevolution. In *Proceedings of the 1998 IEEE World Congress on Computational Intelligence*, pages 570–574, Anchorage, Alaska, USA, 5-9 May 1998. IEEE Press.

[183] M. Quinn. 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, 27-30 2001. IEEE Press. ISBN 0-7803-6658-1.

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

[185] M. Quinn, L. Smith, G. Mayley, and P. Husbands. 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, 2002.

[186] C. Reynolds. An evolved, vision-based behavioral model of coordinated group motion. In *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB92)*, pages 384–392, 1993.

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

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

[189] P. Riley and M. Veloso. On behavior classification in adversarial environments. In L. E. Parker, G. Bekey, and J. Barhen, editors, *Distributed Autonomous Robotic Systems 4*, pages 371–380. Springer-Verlag, 2000.

[190] A. Robinson and L. Spector. Using genetic programming with multiple data types and automatic modularization to evolve decentralized and coordinated navigation in multi-agent systems. In *In Late-Breaking Papers of the Genetic and Evolutionary Computation Conference (GECCO-2002)*. The International Society for Genetic and Evolutionary Computation, 2002.

[191] C. Rosin and R. Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1):1–29, 1997.

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

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

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

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

[196] J. Sauter, R. S. Matthews, H. V. D. Parunak, and S. Brueckner. 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, 2002.

[197] J. Sauter, H. Van Dyke Parunak, S. Brueckner, and R. Matthews. Tuning synthetic pheromones with evolutionary computing. In R. E. Smith, C. Bonacina, C. Hoile, and P. Marrow, editors, *Evolutionary Computation and Multi-Agent Systems (ECOMAS)*, pages 321–324, San Francisco, California, USA, 7 2001.

[198] J. Schmidhuber. Realistic multi-agent reinforcement learning. In *Learning in Distributed Artificial Intelligence Systems, Working Notes of the 1996 ECAI Workshop*, 1996.

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

[200] J. Schneider, W.-K. Wong, A. Moore, and M. Riedmiller. Distributed value functions. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 371–378, 1999.

[201] A. Schultz, J. Grefenstette, and W. Adams. Roboshepherd: Learning complex robotic behaviors. In *Robotics and Manufacturing: Recent Trends in Research and Applications, Volume 6*, pages 763–768. ASME Press, 1996.

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

[203] M. Sekaran and S. Sen. To help or not to help. In *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society*, pages 736–741, Pittsburgh, PA, 1995.

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

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

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

[207] S. Sen and M. Sekaran. Using reciprocity to adapt to others. In G. Weiß and S. Sen, editors, *International Joint Conference on Artificial Intelligence Workshop on Adaptation and Learning in Multiagent Sytems, Lecture Notes in Artificial Intelligence*, pages 206–217. Springer-Verlag, 1995.

[208] S. Sen and M. Sekaran. Multiagent coordination with learning classifier systems. In G. Weiß and S. Sen, editors, *Proceedings of the IJCAI Workshop on Adaption and Learning in Multi-Agent Systems*, volume 1042, pages 218–233. Springer Verlag, 1996. ISBN 3-540-60923-7.

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

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

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

[212] L. Spector and J. Klein. Evolutionary dynamics discovered via visualization in the breve simulation environment. In *Workshop Proceedings of the 8th International Conference on the Simulation and Synthesis of Living Systems*, pages 163–170, 2002.

[213] L. Spector, J. Klein, C. Perry, and M. Feinstein. Emergence of collective behavior in evolving populations of flying agents. In E. Cantu-Paz *et al*, editor, *Prooceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. Springer-Verlag, 2003.

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

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

[216] L. Steels. Emergent adaptive lexicons. In P. Maes, editor, *Proceedings of the Simulation of Adaptive Behavior Conference*. MIT Press, 1996.

[217] L. Steels. Self-organising vocabularies. In *Proceedings of Artificial Life V*, 1996.

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

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

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

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

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

[223] P. Stone. *Layered Learning in Multi-Agent Systems*. PhD thesis, Carnegie Mellon University, 1998.

[224] P. Stone and R. S. Sutton. Keepaway soccer: : A machine learning testbed. In A. Birk, S. Coradeschi, and S. Tadokoro, editors, *RoboCup 2001: Robot Soccer World Cup V*, volume 2377 of *Lecture Notes in Computer Science*, pages 214–223. Springer, 2002. ISBN 3-540-43912-9.

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

[226] D. Subramanian, P. Druschel, and J. Chen. Ants and reinforcement learning: A case study in routing in dynamic networks. In *Proceedings of Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 832–839, 1997.

[227] N. Suematsu and A. Hayashi. 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, 2002.

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

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

[230] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

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

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

[233] G. Tesauro. Temporal difference learning and TD-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[234] S. Thrun. Learning to play the game of chess. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 1069–1076. The MIT Press, Cambridge, MA, 1995.

[235] K. Tumer, A. K. Agogino, and D. H. Wolpert. A bartering approach to improve multiagent learning. In *Proceedings of First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, pages 386–393, 2002.

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

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

[238] J. Vidal and E. Durfee. The moving target function problem in multiagent learning. In *Proceedings of the Third Annual Conference on Multi-Agent Systems*, 1998.

[239] J. M. Vidal and E. H. Durfee. Predicting the expected behavior of agents that learn about agents: the CLRI framework. *Autonomous Agents and Multi-Agent Systems*, January 2003.

[240] K. Wagner. Cooperative strategies and the evolution of communication. *Artificial Life*, 6(2):149–179, Spring 2000.

[241] X. Wang and T. Sandholm. Reinforcement learning to play an optimal Nash equilibrium in team Markov games. In *Advances in Neural Information Processing Systems (NIPS-2002)*, 2002.

[242] R. Watson and J. Pollack. Coevolutionary dynamics in a minimal substrate. In E. Cantu-Paz *et al*, editor, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2001.

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

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

[245] G. Weiß. *Distributed Machine Learning*. Sankt Augustin: Infix Verlag, 1995.

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

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

[248] G. Weiß, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.

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

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

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

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

[253] B. B. Werger and M. J. Mataric. Exploiting embodiment in multi-robot teams. Technical Report IRIS-99-378, University of Southern California, Institute for Robotics and Intelligent Systems, 1999.

[254] G. M. Werner and M. G. Dyer. Evolution of herding behavior in artificial animals. In *From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB92)*, 1993.

[255] T. White, B. Pagurek, and F. Oppacher. ASGA : Improving the ant system by integration with genetic algorithms. In J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 610–617, University of Wisconsin, Madison, Wisconsin, USA, 22-25 1998. Morgan Kaufmann.

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

[257] R. P. Wiegand. *Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, Department of Computer Science, George Mason University, 2003.

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

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

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

[261] M. Wiering, R. Salustowicz, and J. Schmidhuber. Reinforcement learning soccer teams with incomplete world models. *Journal of Autonomous Robots*, 1999.

[262] E. Wilson. *Sociobiology: The New Synthesis*. Belknap Press, 1975.

[263] D. H. Wolpert and K. Tumer. Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(2/3):265–279, 2001.

[264] D. H. Wolpert, K. Tumer, and J. Frank. Using collective intelligence to route internet traffic. In *Advances in Neural Information Processing Systems-11*, pages 952–958, Denver, 1998.

[265] D. H. Wolpert, K. R. Wheller, and K. Tumer. General principles of learning-based multi-agent systems. In O. Etzioni, J. P. Müller, and J. M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 77–83, Seattle, WA, USA, 1999. ACM Press.

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

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

[268] A. Wu, A. Schultz, and A. Agah. Evolving control for distributed micro air vehicles. In *IEEE Computational Intelligence in Robotics and Automation Engineers Conference*, 1999.

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

[270] N. Zaera, D. Cliff, and J. Bruten. (not) evolving collective behaviours in synthetic fish. Technical Report HPL-96-04, Hewlett-Packard Laboratories, 1996.

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

[272] J. Zhao and J. Schmidhuber. Incremental self-improvement for life-time multi-agent reinforcement learning. In P. Maes, M. J. Mataric, J.-A. Meyer, J. Pollack, and S. W. Wilson, editors, *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior: From Animals to Animats 4*, pages 516–525, Cape Code, USA, 9-13 1996. MIT Press. ISBN 0-262-63178-4.