Multiagent Systems: A Survey from a Machine Learning Perspective

Peter Stone and Manuela Veloso

December 4, 1997

CMU-CS-97-193

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Distributed Artificial Intelligence (DAI) has existed as a subfield of AI for less than two decades. DAI is concerned with systems that consist of multiple independent entities that interact in a domain. Traditionally, DAI has been divided into two sub-disciplines: Distributed Problem Solving (DPS) focusses on the information management aspects of systems with several branches working together towards a common goal; Multiagent Systems (MAS) deals with behavior management in collections of several independent entities, or agents. This survey of MAS is intended to serve as an introduction to the field and as an organizational framework. A series of increasingly complex general multiagent scenarios are presented. For each scenario, the issues that arise are described along with a sampling of the techniques that exist to deal with them. The presented techniques are not exhaustive, but they highlight how multiagent systems can be and have been used to build complex systems. When options exist, the techniques presented are biased towards machine learning approaches. Additional opportunities for applying machine learning to MAS are highlighted and robotic soccer is presented as an appropriate testbed for MAS.

This research is sponsored in part by the DARPA/RL Knowledge Based Planning and Scheduling Initiative under grant number F30602-95-1-0018. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of the U. S. Government.

Keywords: Multiagent Systems, survey, Machine Learning, robotic soccer, intelligent agents, pursuit domain, homogeneous agents, heterogeneous agents, communicating agents

1 Introduction

Extending the realm of the social world to include autonomous computer systems has always been an awesome, if not frightening, prospect. However it is now becoming both possible and necessary through advances in the field of Artificial Intelligence (AI). In the past several years, AI techniques have become more and more robust and complex. To mention just one of the many exciting successes, a car recently steered itself more than 95% of the way across the United States using the ALVINN system [60]. By meeting this and other such daunting challenges, AI researchers have earned the right to start examining the implications of multiple autonomous "agents" interacting in the real world. In fact, they have rendered this examination indispensable. If there is one self-steering car, there will surely be more. And although each may be able to drive individually, if several autonomous vehicles meet on the highway, we must know how their behaviors interact.

Multiagent Systems (MAS) is the emerging subfield of AI that aims to provide both principles for construction of complex systems involving multiple agents and mechanisms for coordination of independent agents' behaviors. While there is no generally accepted definition of "agent" in AI [68], for the purposes of this article, we consider an agent to be an entity with goals, actions, and domain knowledge, situated in an environment. The way it acts is called its "behavior." (This is not intended as a general theory of agency.) Although the ability to consider coordinating behaviors of autonomous agents is a new one, the field is advancing quickly by building upon pre-existing work in the field of Distributed Artificial Intelligence (DAI).

DAI has existed as a subfield of AI for less than two decades. Traditionally, DAI is broken into two sub-disciplines: Distributed Problem Solving (DPS) and MAS [10]. The main topics considered in DPS are information management issues such as task decomposition and solution synthesis. For example, a constraint satisfaction problem can often be decomposed into several not entirely independent subproblems that can be solved on different processors. Then these solutions can be synthesized into a solution of the original problem.

MAS allows the subproblems of a constraint satisfaction problem to be subcontracted to different problem solving *agents* with their own interests and goals. Furthermore, domains with multiple agents of any type, including autonomous vehicles and even some human agents, are beginning to be studied.

This survey of MAS is intended as an introduction to the field. The reader should come away with an appreciation for the types of systems that are possible to build using MAS as well as a conceptual framework with which to organize the different types of possible systems.

The article is organized as a series of increasingly complex general multiagent scenarios. For each scenario, the issues that arise are described along with a sampling of the techniques that exist to deal with them. The techniques presented are not exhaustive, but they highlight how multiagent systems can be and have been used to build complex systems.

Because of the inherent complexity of MAS, there is much interest in using machine learning techniques to help deal with this complexity [95, 2]. When several different systems exist that could illustrate the same or similar MAS techniques, the systems presented here are biased towards those that use machine learning (ML) approaches. Furthermore, every effort is made to highlight additional opportunities for applying ML to MAS.

Although there are many possible ways to divide MAS, the survey is organized along two main dimensions: agent heterogeneity and amount of communication among agents. Beginning with the simplest multiagent scenario, homogeneous non-communicating agents, the full range of possible multiagent systems, through highly heterogeneous communicating agents, is considered. Centralized, single-agent systems are shown to belong at the complex end of this spectrum. As illustrated in Figure 1, the heterogeneity dimension is varied first, followed by the communication dimension. The result is a steady increase in system complexity. When appropriate, systems with low agent heterogeneity and high inter-agent communication are also mentioned. However by first increasing heterogeneity and then communication, all of the important issues and techniques in MAS are encountered.

For each multiagent scenario presented, a single example domain is presented in an appropriate instantiation for the purpose of illustration. In this extensively-studied domain, the Predator/Prey or "Pursuit" domain [9], many MAS issues arise. Nevertheless, it is a "toy" domain. At the end of the article, a much more complex domain—robotic soccer—is presented in order to illustrate the full power of MAS.

The article is organized as follows. Section 2 introduces the field of MAS, listing several of its strong points and presenting a taxonomy. The body of the article, Sections 3 – 6, presents the various multiagent scenarios, illustrates them using the pursuit domain, and describes existing work in the field. A domain that facilitates the study of most multiagent issues is advocated as a testbed in Section 7. Section 8 concludes.

2 Multiagent Systems

Two obvious questions about any type of technology are:

- What advantages does it offer over the alternatives?
- In what circumstances is it useful?

It would be foolish to claim that MAS should be used when designing all complex systems. Like any useful approach, there are some situations for which it is particularly appropriate, and others for which it is not. The goal of this section is to underscore the need for and usefulness of MAS while giving characteristics of typical domains that can benefit from it. For a more extensive discussion, see [10].

The most important reason to use MAS when designing a system is that some domains require it. In particular, if there are different people or organizations with different (possibly conflicting) goals and proprietary information, then a multiagent system is needed to handle their interactions. Even if each organization wants to model its internal affairs with a single system, the organizations will not give authority to any single person to build a system that represents them all: the different organizations will need their own systems that reflect their capabilities and priorities.

For example, consider a manufacturing scenario in which company X produces tires, but subcontracts the production of lug-nuts to company Y. In order to build a single system to automate (certain aspects of) the production process, the internals of both companies X and Y must be modeled. However, neither company is likely to want to relinquish information and/or control to a system designer representing the other company. Perhaps with just two companies involved, an agreement could be reached, but with several companies involved, MAS is necessary. The only feasible solution is to allow the various companies to create their own agents that accurately represent their goals and interests. They must then be combined into a multiagent system with the aid of some of the techniques described in this article.

Another example of a domain that requires MAS is hospital scheduling as presented in [20]. This domain from an actual case study requires different agents to represent the interests of different people within the hospital. Hospital employees have different interests, from nurses who want to minimize the patient's time in the hospital, to x-ray operators who want to maximize the throughput on their machines. Since different people evaluate candidate schedules with different criteria, they must be represented by separate agents if their interests are to be justly considered.

Even in domains that could conceivably use systems that are not distributed, there are several possible reasons to use MAS. Having multiple agents could speed up a system's operation by providing a method for parallel computation. For instance, a domain that is easily broken into components—several independent tasks that can be handled by separate agents—could benefit from MAS. Furthermore, the parallelism of MAS can help deal with limitations imposed by time-bounded reasoning requirements.

While parallelism is achieved by assigning different tasks or abilities to different agents, robustness is a benefit of multiagent systems that have redundant agents. If control and responsibilities are sufficiently shared among different agents, the system can tolerate failures by one or more of the agents. Domains that must degrade gracefully are in particular need of this feature of MAS: if a single entity—processor or agent—controls everything, then the entire system could crash if there is a single failure. Although a multiagent system need not be implemented on multiple processors, to provide full robustness against failure, its agents should be distributed across several machines.

Another benefit of multiagent systems is their scalability. Since they are inherently modular, it should be easier to add new agents to a multiagent system than it is to add new capabilities to a monolithic system. Systems whose capabilities and parameters are likely to need to change over time or across agents can also benefit from this advantage of MAS.

From a programmer's perspective the modularity of multiagent systems can lead to simpler programming. Rather than tackling the whole task with a centralized agent, programmers can identify subtasks and assign control of those subtasks to different agents. The difficult problem of splitting a single agent's time among different parts of a task solves itself. Thus, when the choice is between using a multiagent system or a single-agent system, MAS is often the simpler option. Of course there are some domains that are more naturally approached from an omniscient perspective—because a global view is given—or with centralized control—because no parallel actions are possible and there is no action uncertainty [19]. Single-agent systems should be used in such cases.

Finally, multiagent systems can be useful for their illucidation of intelligence [16]. As Gerhard Weiß put it: "Intelligence is deeply and inevitably coupled with interaction" [94]. In fact, it has been proposed that the best way to develop intelligent machines at all might be to start by creating "social" machines [15]. This theory is based on the socio-biological theory that primate intelligence first evolved because of the need to deal with social interactions. Reasons presented above to use MAS are summarized in Table 1.

Table 1: Reasons to use Multiagent Systems

• Some domains require it	• Scalability
• Parallelism	• Simpler programming
• Robustness	• To study intelligence

2.1 Taxonomy

Several taxonomies have been presented previously for the related field of Distributed Artificial Intelligence (DAI). For example, Decker presents four dimensions of DAI [16]:

1. Agent granularity (coarse vs. fine);
2. Heterogeneity of agent knowledge (redundant vs. specialized);
3. Methods of distributing control (benevolent vs. competitive, team vs. hierarchical, static vs. shifting roles);
4. and Communication possibilities (blackboard vs. messages, low-level vs. high-level, content).

Along dimensions 1 and 4, multiagent systems have coarse agent granularity and high-level communication. Along the other dimensions, they can vary across the whole ranges. In fact, the remaining dimensions are very prominent in this article: degree of heterogeneity is a major MAS dimension and all the methods of distributing control appear here as major issues.

More recently, Parunak has presented a taxonomy of MAS from an application perspective [58]. From this perspective, the important characteristics of MAS are:

- System function;
- Agent architecture (degree of heterogeneity, reactive vs. deliberative);
- System architecture (communication, protocols, human involvement).

A useful contribution is that the dimensions are divided into agent and system characteristics. Other overviews of DAI and/or MAS include [47, 23, 25, 10]. This article contributes a taxonomy specifically focussed on MAS along with a detailed chronicle of existing systems as they fit in to this taxonomy.

The taxonomy presented in this article is organized along the most important aspects of agents (as opposed to domains): degree of heterogeneity and degree of communication. Communication is presented as an agent aspect because it is the degree to which the agents communicate (or *whether* they communicate), not the communication protocols that are available to them, that is considered. All the other aspects of agents in MAS are touched upon within the heterogeneity/communication framework. For example, the degree to which different agents play different roles is certainly an important MAS issue, but here it is framed within the scenario of heterogeneous non-communicating agents (it arises in the other two scenarios as well). Domain issues are discussed separately in Section 3.2.

The three combinations of heterogeneity and communication considered in this article (homogeneous non-communicating agents; heterogeneous non-communicating agents; and heterogeneous communicating agents) are presented in order of increasing complexity and power (see Figure 1). Many of the issues that

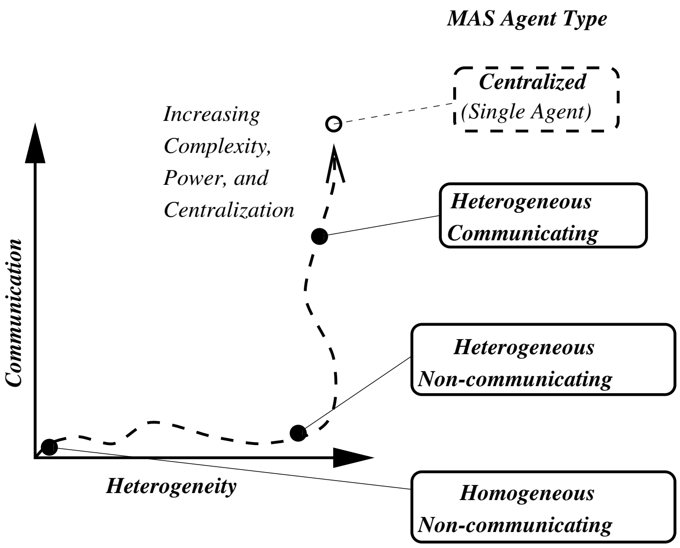

Figure 1: The major categories of the intrafield taxonomy and how they relate to the major dimensions. With full communication of internal state, a centralized system involving a single complex agent may result. Even though there are still multiple entities, they send their sensory perceptions and receive their actions from a central location: a single agent controls them all.

arise in the earlier scenarios also apply in the later scenarios. Nevertheless, they are only mentioned again in the later scenarios to the degree that they differ or become more complex. Notice that single-agent systems are presented as being more complex than multiagent systems. The intuition behind this presentation will become clear in Section 2.2.

The multiagent scenarios along with the issues that arise therein are summarized in Table 2. The techniques that currently exist to address these issues are described in detail in Sections 4 – 6.

Homogeneous Non-communicating Agents

- Reactive vs. deliberative agents
- Local or global perspective
- Modeling of other agents' states
- How to affect others

Heterogeneous Non-communicating Agents

- Benevolence vs. competitiveness
- Evolving agents (arms race, credit/blame)
- Modeling of others' goals, actions, and knowledge
- Resource management (interdependent actions)
- Social conventions
- Roles

Heterogeneous Communicating Agents

- Understanding each other
- Planning communicative acts
- Benevolence vs. competitiveness
- Resource management (schedule coordination)
- Commitment/decommitment
- Truth in communication

Table 2: Issues arising in the various scenarios as reflected in the literature.

2.2 Single-Agent vs. Multiagent Systems

Before studying and categorizing MAS, we must first consider their most obvious alternative: centralized, single-agent systems. Centralized systems have a single agent which makes all the decisions, while the others act as remote slaves. For the purposes of this survey, a "single-agent system" should be thought of as a complex, centralized system in a domain which also allows for a multiagent approach.

A single-agent system might still have multiple entities — several actuators, or even several robots. However, if each entity sends its perceptions to and receives its actions from a single central process, then there is only a single agent: the central process. The central agent models all of the entities as a single "self." This section compares the single-agent and multiagent approaches.

2.2.1 Single-Agent Systems

Although it might seem that single-agent systems should be simpler than multiagent systems, when dealing with a fixed, complex task, the opposite is often the case (see Figure 1). Distributing control among multiple agents allows each agent to be simpler. No one agent has to be able to complete a given task on its own. Thus centralized, single-agent systems belong at the end of the progression from simple to complex multiagent systems in Sections 4 – 6. They are described here first because to many people single-agent (centralized) approaches are more intuitive than multiagent (distributed) ones.

In general, the agent in a single-agent system models itself, the environment, and their interactions. Of course the agent is itself part of the environment, but for the purposes of this article, agents are considered to have extra-environmental components as well. They are independent entities with their own goals, actions, and knowledge. In a single-agent system, no other such entities are recognized by the agent. Thus, even if there are indeed other agents in the world, they are not modeled as having goals, etc.: they are just considered part of the environment. The point being emphasized is that although agents are *also* a part of the environment, they are explicitly modeled as having their own goals, actions, and domain knowledge (see Figure 2).

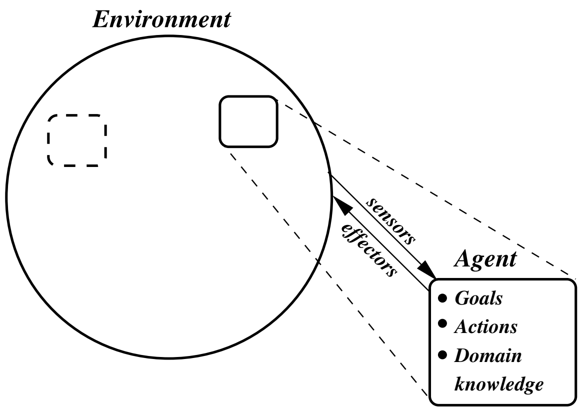

Figure 2: A general single-agent framework. The agent models itself, the environment, and their interactions. If other agents exist, they are considered part of the environment.

2.2.2 Multiagent Systems

Multiagent systems differ from single-agent systems in that several agents exist which model each other's goals and actions. In the fully general multiagent scenario, there may be direct interaction among agents (communication). Although this interaction could be viewed as environmental stimuli, we present inter-agent communication as being separate from the environment.

From an individual agent's perspective, multiagent systems differ from single-agent systems most significantly in that the environment's dynamics can be determined by other agents. In addition to the uncertainty that may be inherent in the domain, other agents intentionally affect the environment in unpredictable ways. Thus, all multiagent systems can be viewed as having dynamic environments.

Figure 3 illustrates the view that each agent is both part of the environment and modeled as a separate entity. There may be any number of agents, with different degrees of heterogeneity and with or without the ability to communicate directly. From the fully general case depicted here, we begin by eliminating both the communication and the heterogeneity to present homogeneous non-communicating MAS (Section 4). Then, in Section 5 the possibility of agent heterogeneity is added back in. Finally, in Section 6, we arrive back at the fully general case by considering agents that can interact directly.

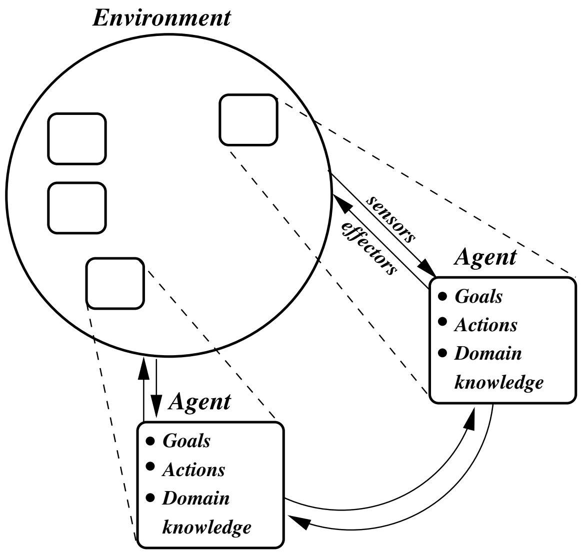

Figure 3: The fully general multiagent scenario. Agents model each other's goals and actions; they may also interact directly (communicate).

3 Organization of Existing Work

The following sections present many different MAS techniques that have been previously published. They present an extensive, but not exhaustive, list of work in the field. Despite the youth of the field, space does not permit exhaustive coverage. Instead, the work mentioned is intended to illustrate the techniques that exist to deal with the issues that arise in the various multiagent scenarios. When possible, ML approaches are emphasized.

In increasing order of complexity, the three multiagent scenarios considered are: homogeneous non-communicating agents, heterogeneous non-communicating agents, and heterogeneous communicating agents. For each of these scenarios, the research issues that arise, the techniques that deal with them, and additional ML opportunities are presented. The issues may appear across scenarios, but they are presented and discussed in the least complex scenario to which they apply.

In addition to the existing learning approaches described in the sections entitled "Issues and Techniques", there are several previously unexplored learning opportunities that apply in each of the multiagent scenarios. For each scenario, a few promising opportunities for ML researchers are presented.

Many existing ML techniques can be directly applied in multiagent scenarios by delimiting a part of the domain that only involves a single agent. However *multiagent learning* is more concerned with learning issues that arise because of the multiagent aspect of a given domain. As described by Weiß, multiagent learning is "learning that is done by several agents and that becomes possible only because several agents are present" [93]. This type of learning is emphasized in the sections entitled "Further Learning Opportunities."

For the purpose of illustration, each scenario is accompanied by a suitable instantiation of the Predator/Prey or "Pursuit" domain.

3.1 The Predator/Prey ("Pursuit") Domain

The Predator/Prey, or "Pursuit" domain (hereafter referred to as the "pursuit domain"), is an appropriate one for illustration of MAS because it has been studied using a wide variety of approaches and because it has many different instantiations that can be used to illustrate different multiagent scenarios. It is not presented as a complex real-world domain, but rather as a toy domain that helps concretize many concepts. For discussion of a domain that has the full range of complexities characteristic of more real-world domains, see Section 7.

The pursuit domain was introduced by Benda et al. [9]. Over the years, researchers have studied several variations of its original formulation. In this section, a single instantiation of the domain is presented. However, care is taken to point out the parameters that can be varied.

The pursuit domain is usually studied with four *predators* and one *prey*. Traditionally, the predators are blue and the prey is red (black and grey respectively in Figure 4). The domain can be varied by using different numbers of predators and prey.

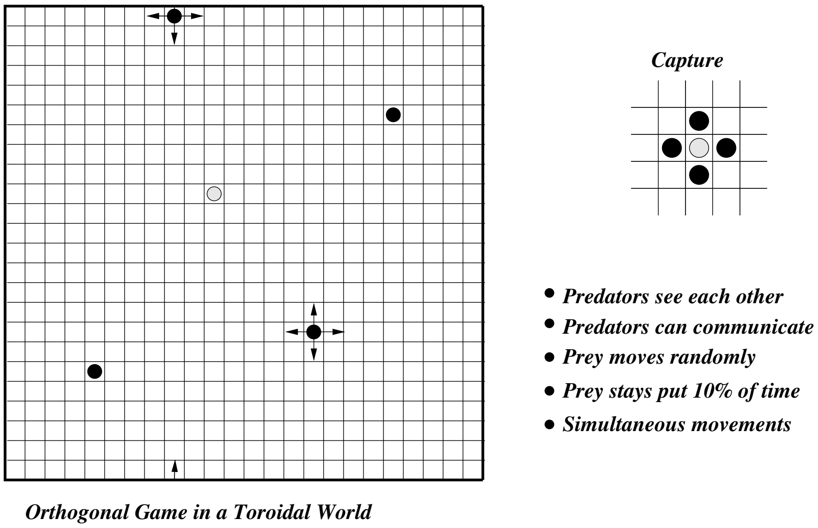

Figure 4: A particular instantiation of the pursuit domain. Predators are black and the prey is grey. The arrows on top of two of the predators indicate possible moves.

The goal of the predators is to "capture" the prey, or surround it so that it cannot move to an unoccupied position. A capture position is shown in Figure 4. If the world has edges, fewer than four predators can capture the prey by trapping it against an edge or in a corner. Another possible criterion for capture is that a predator occupies the same position as the prey. Typically, however, no two players are allowed to occupy the same position.

As depicted in Figure 4, the predators and prey move around in a discrete, grid-like world with square spaces. They can move to any adjacent square on a given turn. Possible variations include grids with other shapes as spaces (for instance hexagons) or continuous worlds. Within the square game, players may be allowed to move diagonally instead of just horizontally. The size of the world may also vary from an infinite plane to a small, finite board with edges. The world pictured in Figure 4 is a toroidal world: the predators and prey can move off one end of the board and come back on the other end. Other parameters of the game that must be specified are whether the players move simultaneously or in turns; how much of the world the predators can see; and whether and how the predators can communicate.

Finally, in the original formulation of the domain, and in most subsequent studies, the prey moves randomly: on each turn it moves in a random direction, staying still with a certain probability in order to simulate being slower than the predators. However, it is also possible to allow the prey to actively try to escape capture. As is discussed in Section 5, there has been some research done to this effect, but there is still much room for improvement. The parameters that can be varied in the pursuit domain are summarized in Table 3.

Table 3: Variable parameters in the pursuit domain

- Definition of capture
- Size and shape of the world
- Legal moves
- Simultaneous or sequential movement
- Visible objects and range
- Predator communication
- Prey movement

The pursuit domain is a good one for the purposes of illustration because it is simple to understand and because it is flexible enough to illustrate a variety of scenarios. The possible actions of the predators and prey are limited and the goal is well-defined. In terms of the reasons to use MAS as presented in Table 1, the pursuit domain does not necessarily require MAS. But in certain instantiations it can make use of the parallelism, robustness, and simpler programming offered by MAS.

In the pursuit domain, a single-agent approach is possible: the agent can observe the positions of all four predators and decide how each of them should move. Since the prey moves randomly rather than intentionally, it is not associated with any agent. Instead it is considered part of the environment as shown in Figure 5. It is also possible to consider DPS approaches to the pursuit domain by breaking the task into subproblems to be solved by each predator. However, most of the approaches described here model the predators as independent agents with a common goal. Thus, they comprise a multiagent system.

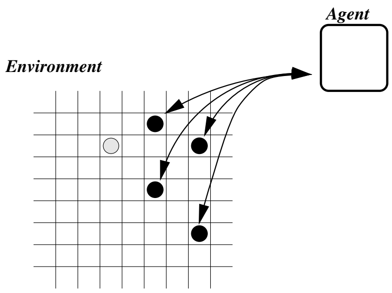

Figure 5: The pursuit domain with just a single agent. One agent controls all predators and the prey is considered part of the environment.

For each of the multiagent scenarios presented below, a new instantiation of the pursuit domain is defined. Their purpose is to illustrate the different scenarios within a concrete framework.

3.2 Domain Issues

Throughout this survey, the focus is upon agent capabilities. However, from the point of view of the system designer, the characteristics of the domain are at least as important. Before moving on to the agent-based categorization of the field, a range of domain characteristics is considered.

Relevant domain characteristics include: the number of agents; the amount of time pressure (is it a real-time domain?); whether or not new goals arrive dynamically; the cost of communication; the cost of failure; user involvement; and environmental uncertainty. The first several of these characteristics are self-explanatory and do not need further mention.

With respect to cost of failure, an example of a domain with high cost of failure is air-traffic control [63]. On the other hand, the directed improvisation domain considered by Hayes-Roth et al. has a very low cost of failure [35]. In this domain, entertainment agents accept all improvisation suggestions from each other. The idea is that the agents should not be afraid to make mistakes, but rather should "just let the words flow" [35].

Several multiagent systems include humans as one or more of the agents. In this case, the designer must consider the issue of communication between the human and computer agents [71]. Another example of user involvement is user feedback in an information filtering domain [27].

Decker distinguishes three different sources of uncertainty in a domain [17]. The transitions in the domain itself might be non-deterministic; agents might not know the actions of other agents; and agents might not know the outcomes of their own actions. This and the other domain characteristics are summarized in Table 4.

Table 4: Domain characteristics that are important when designing MAS

- Number of agents
- Amount of time pressure (real time?)
- Dynamically arriving goals?
- Cost of communication
- Cost of failure
- User involvement
- Environmental uncertainty: *Decker* [17]
  - a priori in the domain
  - in the actions of other agents
  - in outcomes of an agent's own actions

4 Homogeneous Non-Communicating Multiagent Systems

The simplest multiagent scenario involves homogeneous non-communicating agents. In this scenario, all of the agents have the same internal structure including goals, domain knowledge, and *possible* actions. They also have the same procedure for selecting among their actions. The only differences among agents are their sensory inputs and the actual actions they take: they are situated differently in the world.

4.1 Homogeneous Non-Communicating Multiagent Pursuit

In the homogeneous non-communicating version of the pursuit domain, rather than having one agent controlling all four predators, there is one *identical* agent per predator. Although the agents have identical capabilities and decision procedures, they may have limited information about each other's internal state and sensory inputs. Thus they may not be able to predict each other's actions. The pursuit domain with homogeneous agents is illustrated in Figure 6.

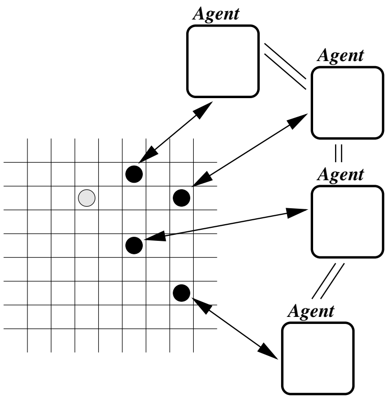

Figure 6: The pursuit domain with homogeneous agents. There is one identical agent per predator. Agents may have (the same amount of) limited information about other agents' internal states.

Within this framework, Stephens and Merx propose a simple heuristic behavior for each agent that is based on local information [81]. They define *capture positions* as the four positions adjacent to the prey. They then propose a "local" strategy whereby each predator agent determines the capture position to which it is closest and moves towards that position. The predators cannot see each other, so they cannot aim at different capture positions. Of course a problem with this heuristic is that two or more predators may move towards the same capture position, blocking each other as they approach. This strategy is not very successful, but it serves as a basis for comparison with two other control strategies—"distributed" and "central"—that are discussed in Section 6.

Since the predators are identical, they can easily predict each other's actions given knowledge of each other's sensory input. Such prediction can be useful when the agents move simultaneously and would like to base their actions on where the other predators will be at the next time step. Vidal and Durfee analyze such a situation using the Recursive Modeling Method (RMM) [90]. RMM is discussed in more detail below, but the basic idea is that predator A bases its move on the predicted move of predator B and vice versa. Since the resulting reasoning can recurse indefinitely, it is important for the agents to bound the amount of reasoning they use either in terms of time or in terms of levels of recursion. Vidal and Durfee's Limited Rationality RMM algorithm is designed to take such considerations into account [90].

Levy and Rosenschein use a game theoretical approach to the pursuit domain [48]. They use a payoff function that allows selfish agents to cooperate. A requirement for their model is that each predator has full information about the location of other predators. Their game model mixes game-theoretical cooperative and non-cooperative games.

Korf also takes the approach that each agent should try to greedily maximize its own local utility [46]. He introduces a policy for each predator based on an attractive force to the prey and a repulsive force from the other predators. Thus the predators tend to approach the prey from different sides. This policy is very successful, especially in the diagonal (agents can move diagonally as well as orthogonally) and hexagonal (hexagonal grid) games. Korf draws the conclusion that explicit cooperation is rarely necessary or useful, at least in the pursuit domain and perhaps more broadly:

> We view this work as additional support for the theory that much coordination and cooperation in both natural and man-made systems can be viewed as an emergent property of the interaction of greedy agents maximizing their particular utility functions in the presence of environmental constraints.
> *Richard Korf* [46]

However, whether or not altruism occurs in nature, there is certainly some use for benevolent agents in MAS, as shown below. More pressingly, if Korf's claim that the pursuit domain is easily solved with local greedy heuristics were true, there would be no point in studying the pursuit domain any further. Fortunately, Haynes and Sen show that Korf's heuristics do not work for certain instantiations of the domain [36] (see Section 5).

4.2 General Homogeneous MAS

The general multiagent scenario with homogeneous agents is illustrated in Figure 7. There are several different agents with identical structure (sensors, effectors, domain knowledge, and decision functions), but they have different sensor input and effector output. That is to say, they are situated differently in the environment and they make their own decisions regarding which actions to take. Having different effector output is a necessary condition for MAS: if the agents all act as a unit, then they are essentially a single agent. In order to realize this difference in output, homogeneous agents must have different sensor input as well. Otherwise they will act identically. For this scenario, in which we consider non-communicating agents, assume that the agents cannot communicate directly.

4.3 Issues and Techniques

Even in this simplest of multiagent scenarios, there are several issues with which to deal. The techniques provided here are representative examples of ways to deal with the presented issues. The issues and techniques, as well as the learning opportunities discussed later, are summarized in Table 5.

4.3.1 Reactive vs. Deliberative agents

When designing any agent-based system, it is important to determine how sophisticated the agents' reasoning will be. Reactive agents simply retrieve pre-set behaviors similar to reflexes without maintaining any internal

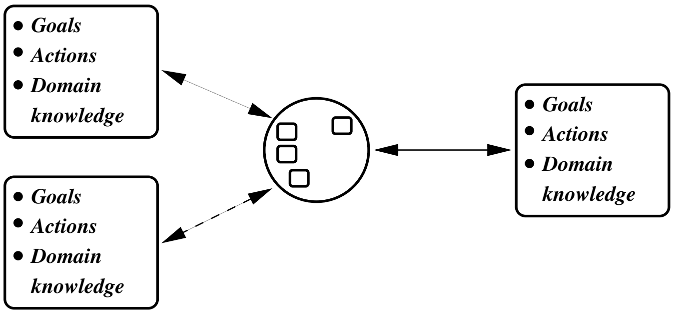

Figure 7: MAS with homogeneous agents. Only the sensor input and effector output of agents differ, as represented by the different arrow styles. The agents' goals, actions, and/or domain knowledge are all identical as indicated by the identical fonts.

Homogeneous Non-Communicating

Issues

- Reactive vs. deliberative agents
- Local or global perspective
- Modeling of other agents' states
- How to affect others

Learning opportunities

- Enable others' actions
- Sensor data → Other agent's sensor data

Techniques

- Reactive Behaviors for Formation maintenance. *Balch* [7]
- Local knowledge sometimes better. *Roychowdhury* [67]
- (limited) Recursive Modeling Method (RMM). *Durfee* [24]
- Don't model others–just pay attention to reward. *Schmidhuber* [77]
- Stigmergy. *Holland/Goldman and Rosenschein* [39, 31]
- Q-learning for behaviors like foraging, homing, etc. *Mataric* [52]

Table 5: The issues, techniques, and learning opportunities for homogeneous MAS as reflected in the literature.

state. On the other hand, deliberative agents behave more like they are thinking, by searching through a space of behaviors, maintaining internal state, and predicting the effects of actions. Although the line between reactive and deliberative agents can be somewhat blurry, an agent with no internal state is certainly reactive, and one which bases its actions on the predicted actions of other agents is deliberative. Here we describe one system at each extreme as well as two others that mix reactive and deliberative reasoning.

Balch and Arkin use homogeneous, reactive, non-communicating agents to study formation maintenance in autonomous robots [7]. The robots' goal is to move together in a military formation such as a diamond, column, or wedge. They periodically come across obstacles which prevent one or more of the robots from moving in a straight line. After passing the obstacle, all robots must adjust in order to regain their formation. The agents reactively convert their sensory data (which includes the positions of the other robots) to motion vectors for avoiding obstacles, avoiding robots, moving to a goal location, and formation maintenance. The actual robot motion is a simple weighted sum of these vectors.

At the other extreme is the pursuit domain work by Levy and Rosenschein that is mentioned above [48]. Their agents assume that each will act in service of its own goals. They use game theoretic techniques to find equilibrium points and thus to decide how to act [48]. These agents are clearly deliberative, as they search for actions rather than simply retrieving them.

There are also several existing systems and techniques that mix reactive and deliberative behaviors. One example is Rao and Georgeff's OASIS system (see Section 6) which reasons about when to be reactive and when to follow goal-directed plans [63]. Another example is Sahota's *reactive deliberation* technique [69]. As the name implies it mixes reactive and deliberative behavior: an agent reasons about which reactive behavior to follow under the constraint that it must choose actions at a rate of 60 Hz. Reactive deliberation was not explicitly designed for MAS, but because it was designed for real-time control in dynamic environments, it is likely to be extendible to multiagent scenarios.

4.3.2 Local or global perspective

Another issue to consider when building a multiagent system is how much sensor information should be available to the agents. Even if it is feasible within the domain to give the agents a global perspectives of the world, it may be more effective to limit them to local views.

Roychowdhury et al. consider a case of multiple agents sharing a set of identical resources in which they have to learn (adapt) their resource usage policies [67]. Since the agents are identical and do not communicate, if they all have a global view of the current resource usage, they will all move simultaneously to the most under-used resource. However, if they each see a partial picture of the world, then different agents gravitate towards different resources: a preferable effect. Better performance by agents with less knowledge is occasionally summarized by the cliche "Ignorance is Bliss."

4.3.3 Modeling of other agents' states

Durfee gives another example of "Blissful Ignorance," mentioning it explicitly in the title of his paper: "Blissful Ignorance: Knowing Just Enough to Coordinate Well" [24]. Now rather than referring to resource usage, the saying applies to the limited Recursive Modeling Method (RMM). As mentioned above in the context of the pursuit domain, RMM could recurse indefinitely. Even if further information can be obtained by reasoning about what agent A thinks agent B thinks agent A thinks . . . , endless reasoning can lead to inaction. Durfee contends that for coordination to be possible, some potential knowledge must be ignored. As well as illustrating this concept in the pursuit domain [90], Durfee goes into more detail and offers more generally applicable methodology in [24].

The point of the RMM is to model the internal state of another agent in order to predict its actions. Even though the agents know each other's goals and structure (they are homogeneous), they may not know each other's future actions. The missing pieces of information are the internal states (for deliberative agents) and sensory inputs of the other agents. How and whether to model other agents is a ubiquitous issue in MAS. In the more complex multiagent scenarios presented in the next sections, agents may have to model not only the internal states of other agents, but also their goals, actions, and abilities.

Although it may be useful to build models of other agents in the environment, agent modeling is not done universally. Schmidhuber advocates a form of multiagent reinforcement learning (RL) with which agents do not model each other as agents [77]. Instead they consider each other as parts of the environment and affect each other's policies only as sensed objects. The agents pay attention to the reward they receive using a given policy and checkpoint their policies so they can return to successful ones. Schmidhuber shows that the agents can learn to cooperate without modeling each other.

4.3.4 How to affect others

When no communication is possible, system designers must decide how the agents will affect one another. Since they exist in the same environment, the agents can affect each other in several ways. Actively, they can be sensed by other agents, or they may be able to change the state of another agent by, for example, pushing it. More indirectly, agents can affect other agents by one of two types of *stigmergy* [39]. First, *active stigmergy* occurs when an agent alters the environment so as to affect the sensory input of another agent. For example, a robotic agent might leave a marker behind it for other agents to observe. Goldman and Rosenschein demonstrate an effective form of active stigmergy in which agents heuristically alter the environment in order to facilitate future unknown plans of other agents [31]. Second, *passive stigmergy* involves altering the environment so that the effects of another agent's actions change. For example, if one agent turns off the main water valve to a building, the effect of another agent turning on the kitchen faucet is altered.

Holland illustrates the concept of passive stigmergy with a robotic system designed to model the behavior of an ant colony confronted with many dead ants around its nest [39]. An ant from such a colony tends to periodically pick up a dead ant, carry it for a short distance, and then drop it. Although the behavior appears to be random, after several hours, the dead ants are clustered in a small number of heaps. Over time, there are fewer and fewer large piles until all the dead ants end up in one pile. Although the ants behave homogeneously and, at least in this case, we have no evidence that they communicate explicitly, the ants manage to cooperate in achieving a task.

Holland models this situation with a number of identical robots in a small area scattered with pucks [39]. The robots are programmed reactively to move straight (turning at walls) until they are pushing three or more pucks. At that point, the robots back up and turn away, leaving the three pucks in a cluster. Although the robots do not communicate at all, they are able to collect the pucks into a single pile over time. This effect occurs because when a robot approaches an existing pile directly, it adds the pucks it was already carrying to the pile and turns away. Of course a robot approaching an existing pile obliquely might take a puck away from the pile, but over time the desired result is accomplished. Like the ants, the robots use passive stigmergy to affect each other's behavior.

A similar scenario with more deliberative robots is explored by Mataric. In this case, the robots use Q-learning to learn behaviors including foraging for pucks as well as homing and following [52]. The robots learn independent policies, dealing with the high-dimensional state space with the aid of *progress estimators* that give intermediate rewards, and with the aid of boolean value predicates that condense many states into one. Mataric's robots actively affect each other through observation: a robot learning to follow another robot can base its action on the relative location of the other robot.

4.4 Further Learning Opportunities

In addition to the existing learning approaches described above, there are several previously unexplored learning opportunities that apply to homogeneous non-communicating systems (see Table 5).

One unexplored learning opportunity that could apply in domains with homogeneous non-communicating agents is learning to enable others' actions. Inspired by the concept of stigmergy, an agent may try to learn to take actions that will not directly help it in its current situation, but that may allow other similar agents to be more effective in the future. Typical RL situations with delayed reward encourage agents to learn to achieve their goals directly by propagating local reinforcement back to past states and actions [42]. However if an action leads to a reward by another agent, the acting agent may have no way of reinforcing that action. Techniques to deal with such a problem would be useful for building multiagent systems.

In terms of modeling other agents, there is much room for improvement in the situation that a given agent does not know the internal state or sensory inputs of another agent. When such information is known, RMM can be used to determine future actions of agents. However, if the information is not directly available, it would be useful for an agent to learn it. The function from agent X's sensor data (which might include a restricted view of agent Y) to agent Y's sensor data is a useful function to learn. If effectively learned, agent X can then use (limited) RMM to predict agent Y's future actions.

5 Heterogeneous Non-Communicating Multiagent Systems

To this point, we have only considered agents that are homogeneous. Adding the possibility of heterogeneous agents in a multiagent domain adds a great deal of potential power at the price of added complexity. Agents might be heterogeneous in any of a number of ways, from having different goals to having different domain models and actions. An important subdimension of heterogeneous agent systems is whether agents are benevolent or competitive. Even if they have different goals, they may be friendly to each other's goals or they may actively try to inhibit each other. This aspect of heterogeneous systems, along with several others, is described below.

5.1 Heterogeneous Non-Communicating Multiagent Pursuit

Before exploring the general multiagent scenario involving heterogeneous non-communicating agents, consider how this scenario can be instantiated in the pursuit domain. As in the previous scenario, the predators are controlled by separate agents. But they are no longer necessarily identical agents: their goals, actions and domain knowledge may differ. In addition, the prey, which inherently has goals different from those of the predators, can now be modeled as an agent. The pursuit domain with heterogeneous agents is shown in Figure 8.

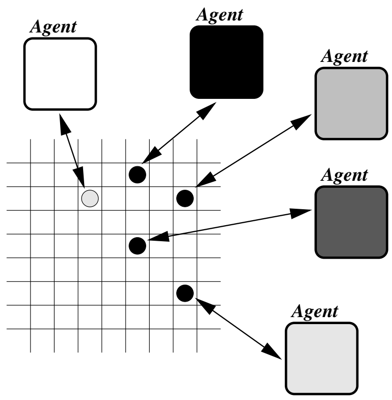

Figure 8: The pursuit domain with heterogeneous agents. Goals and actions may differ among agents. Now the prey may also be modeled as an agent.

Haynes and colleagues have done various studies with heterogeneous agents in the pursuit domain. They have evolved teams of predators, equipped predators with case bases, and competitively evolved the predators and the prey.

First, Haynes et al. use genetic programming (GP) to evolve teams of four predators [38]. Rather than evolving predator agents in a single evolutionary pool and then combining them into teams to test performance, each individual in the population is actually a team of four agents already specifically assigned to different predators. Thus the predators can evolve to cooperate. This co-evolution of teammates is one possible way around the absence of communication in a domain. In place of communicating planned actions to each other, the predators can evolve to *know*, or at least act as if knowing, each other's future actions.

In a separate study, Haynes et al. use case-based reasoning to allow predators to learn to cooperate [36]. They begin with identical agents controlling each of the predators. The predators move simultaneously to their closest capture positions. But because predators that try to occupy the same position all remain stationary, cases of deadlock arise. When deadlock occurs, the agents store the negative case so as to avoid it in the future, and they try different actions. Keeping track of which agents act in which way for given deadlock situations, the predators build up different case bases and thus become heterogeneous agents. Over time, the predators learn to stay out of each other's way while approaching the prey.

Finally, Haynes and Sen explore the possibility of evolving both the predators and the prey so that they all try to improve their behaviors [37]. Working in a toroidal world and starting with predator behaviors such as Korf's greedy heuristic and their own evolved GP predators, they then evolve the prey to behave more effectively than randomly. Although one might think that continuing this process would lead to repeated improvement of the predator and prey behaviors with no convergence, a prey behavior emerges that always succeeds: the prey simply moves in a constant straight line. Even when allowed to re-adjust to the "linear" prey behavior, the predators are unable to reliably capture the prey. Haynes and Sen conclude that Korf's greedy solution to the pursuit domain relies on random prey movement which guarantees locality of movement. Although there may yet be greedy solutions that can deal with different types of prey behavior, they have not yet been discovered. Thus the predator domain retains value for researchers in MAS.

Although Haynes and Sen convince the reader that the pursuit domain is still worth studying [37], the co-evolutionary results are less than satisfying. As mentioned above, one would intuitively expect the predators to be able to adapt to the linearly moving prey. For example, since they operate in a toroidal world, a single predator could place itself in the prey's line of movement and remain still. Then the remaining predators could surround the prey at their leisure. The fact that the predators are unable to re-evolve to find such a solution suggests that either the predator evolution is not performed optimally, or slightly more "capable" agents (i.e. agents able to reason more about past world states) would lead to a more interesting study. Nevertheless, the study of competitive co-evolution in the pursuit domain started by Haynes and Sen is an intriguing open issue.

5.2 General Heterogeneous MAS

The general multiagent scenario with heterogeneous non-communicating agents is depicted in Figure 9. As in the homogeneous case (see Figure 7), the agents are situated differently in the environment which causes them to have different sensory inputs and necessitates their taking different actions. However in this scenario, the agents have much more significant differences. They may have different goals, actions, and/or domain knowledge. This condition of heterogeneity among agents adds a great deal of power for the system designer. In order to focus on the benefits (and complexity) of heterogeneity, the assumption of no communication is retained for this section.

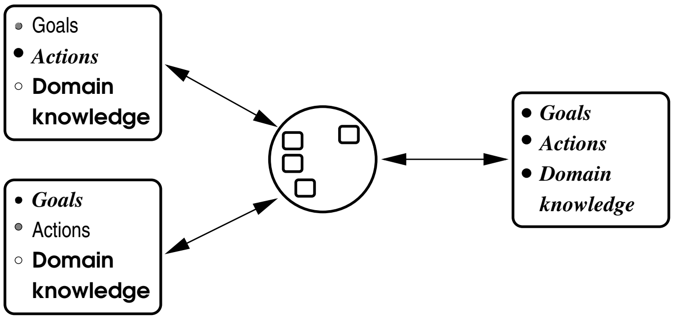

Figure 9: The general heterogeneous MAS scenario. Now agents' goals, actions, and/or domain knowledge may differ as indicated by the different fonts. The assumption of no direct interaction remains.

5.3 Issues and Techniques

Even without communication, numerous issues that were not present in the homogeneous agent scenario (Section 4) arise in this scenario. Some have already been touched upon above in the context of the pursuit domain. These issues and existing techniques to deal with them, along with further learning opportunities, are described below and summarized in Table 6.

5.3.1 Benevolence vs. competitiveness

One of the most important issues to consider when designing a multiagent system is whether the different agents will be benevolent or competitive. Even if they have different goals, the agents can be benevolent if they are willing to help each other achieve their respective goals [31]. On the other hand, the agents may be selfish and only consider their own goals when acting. In the extreme, the agents may be involved in a zero-sum situation so that they must actively oppose other agents' goals in order to achieve their own.

Some people only consider using selfish agents, claiming that they are both more effective when building real systems and more biologically plausible. Of course if agents have the same goals, they will help each other, but people rarely consider agents that help each other achieve different goals for no apparent reason: when agents cooperate, they usually do so because it is in their own best interest. As we have already seen in the pursuit domain, Korf advocates using greedy agents that minimize their own distance to the prey [46], and similarly, Levy and Rosenschein use Game Theory to study how the predators can cooperate despite maximizing their own utilities [48]. Some advocates of selfish agents point to nature for their justification, claiming that animals are not altruistic, but rather act always in their own self-interest [46]. On

Heterogeneous Non-Communicating

Issues

- Benevolence vs. competitiveness
- Stable vs. evolving agents (arms race, credit/blame)
- Modeling of others' goals, actions, and knowledge
- Resource management (interdependent actions)
- Social conventions
- Roles

Learning opportunities

- Credit/blame in competitive scenarios
- Behaviors that blend well with team
- Prediction of others' actions
- Dynamic role assumption

Techniques

- Game theory, iterative play. *Mor and Rosenschein/Sandholm and Crites* [55, 75]
- Minimax-Q. *Littman* [49]
- Competitive co-evolution. *Haynes and Sen/Grefenstette and Daley/Rosin and Belew* [37, 32, 66]
- Deduce intentions through observation. *Huber and Durfee* [40]
- Autoepistemic reasoning (ignorance). *Permpoontanalarp* [59]
- Model as a team (individual $\rightarrow$ role). *Tambe* [85, 86]
- Social reasoning: depend on others for goal ($\neq$ game theory). *Sichman and Demazeau* [79]
- GAs to deal with Braes' paradox (more resource $\rightarrow$ worse). *Arora and Sen* [4]
- Multiagent RL for adaptive Load Balancing. *Schaerf, Shoham, and Tennenholtz* [76]
- Focal points/Emergent conventions. *Fenster et al./Walker and Woolridge* [26, 91]
- Design agents play different roles. *Prasad et al.* [62]

Table 6: The issues, techniques, and learning opportunities for heterogeneous MAS as reflected in the literature.

the other hand, Ridley provides a detailed chronicle and explanation of apparent altruism in nature (usually explainable as kin selection) and cooperation in human societies [64].

Whether or not altruism exists, in some situations it may be in an animal's (or agent's) interest to cooperate with other agents. Mor and Rosenschein illustrate this possibility in the context of the prisoner's dilemma [55]. In the prisoner's dilemma, two agents try to act so as to maximize their own individual rewards. They are not actively out to thwart each other since it is not a zero-sum game, yet they place no inherent value on the other receiving reward. The prisoner's dilemma is constructed so that each agent is given two choices: defect or cooperate. No matter what the other agent does, a given agent receives a higher reward if it defects. Yet if both agents cooperate, they are better off than if they both defect. In any given play, an agent is better off defecting. Nevertheless, Mor and Rosenschein show that if the same agents come up against each other repeatedly (iterated prisoner's dilemma), cooperative behavior can emerge. In effect, an agent can serve its own self-interest by establishing a *reputation* for being cooperative. Then when coming up against another cooperative agent, the two can benefit from a sense of trust for each other: they both cooperate rather than both defecting. Only with repeated play can cooperation emerge among the selfish agents in the prisoner's dilemma.

In the prisoner's dilemma, the agents are selfish but not inherently competitive: in specific circumstances, they are willing to act benevolently. However, when the agents are actually competitive (such as in zero-sum games), cooperation is no longer sensible. For instance, Littman considers a zero-sum game in which two players try to reach opposite ends of a small discrete world. The players can block each other by trying to move to the same space. Littman introduces a variant of Q-learning called Minimax-Q which is designed to work on Markov games as opposed to Markov Decision Processes [49]. The competitive agents learn probabilistic policies since any deterministic policy can be completely counteracted by the opponent.

The issue of benevolence (willingness to cooperate) vs. competitiveness comes up repeatedly in the systems described below. Were a third dimension to be added to the categorization of MAS (in addition to degrees of heterogeneity and communication), this issue would be it.

5.3.2 Stable vs. evolving agents

Another important characteristic to consider when designing multiagent systems is whether the agents are stable or evolving. Of course evolving agents can be useful in dynamic environments. But particularly when using competitive agents, allowing them to evolve can lead to complications. Such systems that use competitive evolving agents are said to use a technique called *competitive co-evolution*. Systems that evolve benevolent agents are said to use *cooperative co-evolution*. The evolution of both predator and prey agents by Haynes and Sen [37] qualifies as competitive co-evolution.

Grefenstette and Daley conduct a preliminary study of competitive and cooperative co-evolution in a domain that is loosely related to the pursuit domain [32]. Their domain has two robots that can move continuously and one morsel of (stationary) food that appears randomly in the world. In the cooperative task, both robots must be at the food in order to "capture" it. Since the robots can run out of energy if they move too much, they learn to move towards food only when both of them are near enough to reach it. Evolving populations of decision rules using Genetic Algorithms (GAs), Grefenstette and Daley consider different methods of fitness evaluation. Fitness evaluation—the evaluation of relative "fitness" of individuals in a population so that the most fit can be retained and recombined—is an important component of evolutionary learning techniques. Grefenstette and Daley find that an effective method for cooperative co-evolution in their domain is to use separate GAs to evolve rules for the two agents, evaluating individuals against a "champion" (individual with highest fitness) from a random generation of the other GA.

In a competitive task in the same domain, agents try to be the first to reach the food [32]. Again, different GA evaluation methods are considered for use in evolving rule sets to control the agents.

One problem to contend with in competitive rather than cooperative co-evolution is the possibility of an escalating "arms race" with no end. Competing agents might continually adapt to each other in more and more specialized ways, never stabilizing at a good behavior. Of course in a dynamic environment, it may not be feasible or even desirable to evolve a stable behavior. Applying RL to the iterated prisoner's dilemma, Sandholm and Crites find that a learning agent is able to perform optimally against a fixed opponent [75]. But when both agents are learning, there is no stable solution.

Another issue in competitive co-evolution is the credit/blame assignment problem. When performance of an agent improves, it is not necessarily clear whether the improvement is due to an improvement in that agent's behavior or a negative change in the opponent's behavior. Similarly, if an agent's performance gets worse, the blame or credit could belong to that agent or to the opponent.

One way to deal with the credit/blame problem is to fix one agent while evolving the other and then switch. Of course this method encourages the arms race more than ever. Nevertheless, Rosin and Belew use this technique, along with an interesting method for maintaining diversity in genetic populations, to evolve agents that can play TicTacToe, Nim, and a simple version of Go [66]. When it is a given agent's turn to evolve, it executes a standard GA generation. Individuals are tested against individuals from the competing population, but a technique called "competitive fitness sharing" is used to maintain diversity. When using this technique, individuals from agent X's population are given more credit for beating opponents (individuals from agent Y's population) that are not beaten by other individuals from agent X's population. More specifically, the reward to an individual for beating individual y is divided by the number of other individuals in agent X's population that also beat individual y. Competitive fitness sharing shows much promise for people building systems that use competitive co-evolution.

5.3.3 Modeling of others' goals, actions, and knowledge

In the case of homogeneous agents, it was useful for agents to model the internal states of other agents in order to predict their actions. With heterogeneous agents, the problem of modeling others is much more complex. Now the goals, actions, and domain knowledge of the other agents may also be unknown and thus need modeling.

Without communication, agents are forced to model each other strictly through observation. Huber and Durfee consider a case of coordinated motion control among multiple mobile robots under the assumption that communication is prohibitively expensive [40]. Thus the agents try to deduce each other's plans by observing their actions. In particular, each robot (simulated or real) tries to figure out the destinations of the other robots by watching how they move. Plan recognition of this type is also useful in competitive domains, since knowing an opponent's goals or intentions can make it significantly easier to defeat.

In addition to modeling agents' goals through observation, it is also possible to learn their actions. Wang's OBSERVER system allows an agent to incrementally learn the preconditions and effects of planning actions by observing domain experts [92]. After observing for a time, the agent can then experimentally refine its model by practicing the actions itself.

When modeling other agents, it may be useful to reason not only about what is true and what is false, but also about what is not known. Such reasoning about ignorances is called *autoepistemic reasoning*. For a theoretical presentation of an autoepistemic reasoning method in MAS, see [59].

Just as RMM is useful for modeling the states of homogeneous agents, it can be used in the heterogeneous scenario as well. Tambe takes it one step further, studying how agents can *learn* models of teams of agents. In an air combat domain, agents can use RMM to try to deduce an opponents' plan based on its observable actions [85]. For example, a fired missile may not be visible, but the observation of a preparatory maneuver commonly used before firing could indicate that a missile has been launched.

When teams of agents are involved, the situation becomes more complicated. In this case, an opponent's actions may not make sense except in the context of a team maneuver. Then the agent's *role* within the team must be modeled. Tambe discusses the advantages of team modeling [86].

One reason that modeling other agents might be useful is that agents sometimes depend on each other for achieving their goals. Unlike in game theory where agents can cooperate or not depending on their utility estimation, there may be actions that *require* cooperation for successful execution. For example, two robots may be needed to successfully push a box, or, as in the pursuit domain, several agents may be needed to capture an opponent. Sichman and Demazeau analyze how the case of conflicting mutual models of different co-dependent agents can arise and be dealt with [79].

5.3.4 Resource management

Heterogeneous agents may have interdependent actions due to limited resources needed by several of the agents. Example domains include network traffic problems in which several different agents must send information through the same network; and load balancing in which several computer processes or users have a limited amount of computing power to share among them. Designers of multiagent systems with limited resources must decide how the agents will share the resources.

One interesting network traffic problem called Braess' paradox has been studied from a multiagent perspective using GAs [30]. Braess' paradox is the phenomenon of adding more resources to a network but getting worse performance. Glance and Hogg claim that under certain conditions, including usage-dependent resource costs, agents that are sharing the network and reasoning separately about which path of the network to use cannot achieve global optimal performance [30]. Glance and Hogg use GAs to represent different parts of a contrived sample network. Arora and Sen then improve the GA representation slightly and show that with the new representation, the system is actually able to find the globally optimal traffic flow [4].

Adaptive load balancing has been studied as a multiagent problem by allowing different agents to decide which processor to use at a given time. Using RL, Schaerf et al. show that the heterogeneous agents can achieve reasonable load balance without any central control and without communication among agents [76]. The agents keep track of how long a job takes when it is scheduled on a given resource, and they are given some incentive to explore untried processors or processors that did poorly in the past.

5.3.5 Social conventions

Although the current multiagent scenario does not allow for communication, there has been some very interesting work done on how heterogeneous agents can nonetheless reach "agreements," or make coinciding choices, if necessary. Humans are able to reach tacit agreements as illustrated by the following scenario:

> Imagine that you and a friend need to meet today. You both arrived in Paris yesterday, but you were unable to get in touch to set a time and place. Nevertheless, it is essential that you meet today. Where will you go, and when?

Rick Vohra posed this question to an audience of roughly 40 people at the AAAI-95 Fall Symposium on Active Learning: roughly 75% of the people wrote down (with no prior communication) that they would go to the Eifel tower at noon. Thus even without communicating, people are sometimes able to coordinate actions. Apparently features that have been seen or used often present themselves as obvious choices.

In the context of MAS, Fenster et al. define the Focal Point method [26]. They discuss the phenomenon of cultural (or programmed) preferences allowing agents to "meet" without communicating. They propose that, all else being equal, agents who need to meet should choose rare or extreme options.

On a similar note, conventions might emerge over time. Walker and Woolridge propose biasing agents towards options that have been chosen, for example, most recently or most frequently in the past [91]. Rather than coming from pre-analysis of the options as in the Focal Point method, conventions emerge over time.

5.3.6 Roles

When agents have similar goals, they can be organized into a team. Each agent then plays a separate *role* within the team. With such a benevolent team of agents, one must provide some method for assigning different agents to different roles. This assignment might be obvious if the agents are very specific and can each only do one thing. However in some domains, the agents are flexible enough to interchange roles.

The multiagent design of a steam pump is one such domain. Prasad et al. study design agents that can either initiate a design or extend a design [62]. In different situations, different agents are more effective at initiation and at extension. Thus a supervised learning technique is used to help agents learn what roles they should fill in different situations.

Although already mentioned above in the context of modeling other agents, Tambe's work deserves mention in this context as well. When an agent is faced with an opposing team of agents, it may be useful to model individual agents as filling roles within a team action rather than as acting independently [86].

5.4 Further Learning Opportunities

Throughout the above investigation of issues and techniques in the heterogeneous non-communicating multiagent scenario, many learning approaches are described. A few of the other most obvious future ML applications to this scenario are described here and summarized in Table 6.

One challenge for system builders who use evolving agents is dealing with the credit/blame problem. When several different agents are evolving at the same time, changes in an agent's fitness could be due to its own behavior or due to the behavior of others. Yet if agents are to evolve effectively, they must have a reasonable idea of whether a given change in behavior is beneficial or detrimental. Methods of objective fitness measurement are also needed for testing various evolution techniques. In competitive (especially zero-sum) situations, it is difficult to provide adequate performance measurements over time. Even if all agents improve drastically, if they all improve the same amount, the actual results could remain the same. One possible way around this problem is to test agents against past agents in order to measure improvement. However this solution is not ideal: the current agent may have adapted to the current opponent rather than past opponents. A reliable measurement method would be a valuable contribution to ML in MAS.

In cooperative situations, agents ideally learn to behave in such a way that they can help each other. Unfortunately, most existing ML techniques focus on exploring behaviors that are likely to help an agent with its own "personal" deficiencies. An interesting contribution would be a method for introducing into the learning space a bias towards behaviors that are likely to blend well with the behaviors of other agents.

Many of the techniques described in this section pertained to modeling other agents in the heterogeneous non-communicating scenario. However the true end is not just knowledge of another agent's current situation, but rather the ability to predict its future actions. For example, the reason it is useful to deduce another mobile robot's goal location is that its path to the goal may then be predicted and collision avoided. There is still much room for improvement of existing techniques and for new techniques that allow agents to predict each other's future actions.

In the context of teams of agents, it has been mentioned that agents might be suited to different roles in different situations. In a dynamic environment, these flexible agents are more effective if they can switch roles dynamically. For example, if an agent finds itself in a position to easily perform a useful action that is not usually considered a part of its current role, it may switch roles and leave its old role available for another agent. A challenging possible approach to this problem is to enable the agents to learn which roles they should assume in what situations. Dynamic role assumption is a particularly good opportunity for ML researchers in MAS.

6 Heterogeneous Communicating Multiagent Systems

The scenarios examined thus far have included agents that differ in any number of ways, including their sensory data, their goals, their actions, and their domain knowledge. Such heterogeneous multiagent systems can be very complex and powerful. However the full power of MAS can be realized when adding the ability for agents to communicate with one another. In fact, adding communication introduces the possibility of having a multiagent system turn into a system that is essentially equivalent to a single-agent system. By sending their sensor inputs to and receiving their commands from one agent, all the other agents can surrender control to that single agent. In this case, control is no longer distributed. Thus communicating heterogeneous agents can span the full range of complexity in agent systems.

Admittedly, communication could be viewed as simply part of an agent's interaction with the environment. However just as agents are considered special parts of the environment for the purposes of this survey, so is communication among agents considered extra-environmental. With the aid of communication, agents can coordinate much more effectively than they have been able to up to this point. In this scenario we include homogeneous as well as heterogeneous communicating agents.

6.1 Heterogeneous Communicating Multiagent Pursuit

In the pursuit domain, communication creates new possibilities for predator behavior. Here, agents can still be fully heterogeneous. But now cooperating agents can also communicate with one another. Since the prey acts on its own in the pursuit domain, it has no other agents with which to communicate. However the predators can freely exchange information in order to help them capture the prey more effectively. The current situation is illustrated in Figure 10.

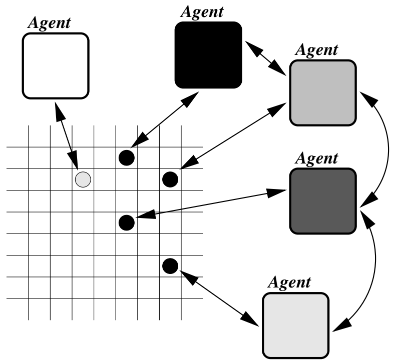

Figure 10: The pursuit domain with communicating agents. Agents can still be fully heterogeneous but now the predators can communicate with one another.

Tan uses communicating agents in the pursuit domain to conduct some interesting multiagent Q-learning experiments [87]. In his instantiation of the domain, there are several prey agents and the predators have limited vision so that they may not always know where the prey are. Thus the predators can help each other by informing each other of their sensory input. Tan shows that they might also help each other by exchanging reinforcement episodes and/or control policies.

Recall the "local" strategy defined by Stephens and Merx in which each predator simply moved to its closest "capture position." In their instantiation of the domain, the predators can see the prey, but not each other. With communication possible, they define two more possible strategies for the predators [81]. When using a "distributed" strategy, the agents are still homogeneous, but they communicate to insure that each moves toward a different capture position. In particular, the predator farthest from the prey chooses the capture position closest to it, and announces that it will approach that position. Then the next farthest predator chooses the closest capture position from the remaining three, and so on. This simple protocol encourages the predators to close in on the prey from different sides. A distributed strategy, it is much more effective than the local policy and does not require very much communication. However there are situations in which it does not succeed.

Stephens and Merx then present one more strategy that always succeeds but requires much more communication: the "central" strategy [81]. The central strategy is effectively a single agent system. Three predators transmit all of their sensory inputs to one central agent which then decides where all the predators should move and transmits its decision back to them. In this case, there is really only one intelligent controlling agent and three puppets. Observe that by taking MAS to the extreme of full communication, we may arrive at a single-agent system.

Benda et al., in the original presentation of the pursuit domain, also consider the full range of communication possibilities, all the way up to the central strategy [9]. They consider the possible organizations of the four predators when any pair can either exchange data, exchange data and goals, or have one control the other. The tradeoff between lower communication costs and better decisions is described. Communication costs might come in the form of limited bandwidth or consumption of reasoning time.

Another way to frame this tradeoff is as one between cost and freedom: as communication cost (time) increases, freedom decreases. Osawa suggests that the predators should move through four phases. In increasing order of cost (decreasing freedom), they are: autonomy, communication, negotiation, and control [57]. When the predators stop making sufficient progress toward the prey using one strategy, they should move to the next most expensive strategy. Thus they can close in on the prey efficiently and effectively.

We identify an important lesson to learn from the above examples:

In terms of increasing complexity, the extreme multiagent scenario is a complex single-agent scenario.

6.2 General Communicating MAS

Indeed, this continuum of complexity leading into the extreme single agent case applies for MAS in general (see Figure 1). With communicating agents, systems can get arbitrarily complex and arbitrarily centralized until a single agent has all the control. Of course communication bandwidth may be prohibitively low to reach the extreme in a given domain.

The fully general multiagent scenario appears in Figure 11. In this scenario, we allow the agents to be heterogeneous to any degree from homogeneity to full heterogeneity. The key addition is the ability for agents to transmit information directly to each other. From a practical point of view, the communication might be broadcast or posted on a "blackboard" for all to interpret, or it might be targeted point-to-point from an agent to another specific agent.

6.3 Issues and Techniques

Since heterogeneous communicating agents can choose not to communicate, and in some cases can also choose to be homogeneous or at least to minimize their heterogeneity, most of the issues discussed in the previous two scenarios apply in this one as well. But the ability to communicate raises another whole set of issues for which techniques exist. Two of the most studied issues are communication protocols and theories of commitment. The issue of benevolence vs. competitiveness, already discussed in the previous MAS scenario, becomes more complicated in this context. These issues and others along with some of the existing techniques to deal with them and further learning opportunities are described below and summarized in Table 7.

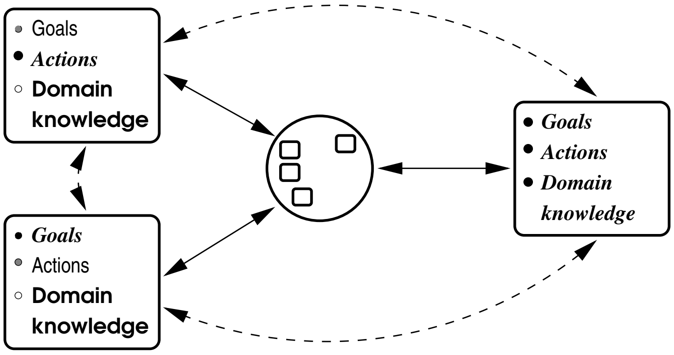

Figure 11: The general communicating MAS scenario. Agents can be heterogeneous to any degree. Information can be transmitted directly among agents as indicated by the arrows between agents. Communication can either be broadcast or transmitted point-to-point.

Heterogeneous Communicating

Issues

- Understanding each other
- Planning communicative acts
- Benevolence vs. competitiveness
- Resource management (schedule coordination)
- Commitment/decommitment
- Truth in communication

Learning opportunities

- Evolving language
- Effects of speech acts on global dynamics
- Communication utility and truthfulness
- Commitment utility

Techniques

- Language protocols: KIF for content (*Genesreth and Fikes* [29]),
  KQML for message format (*Finin et al.* [28]).
- Speech acts. *Cohen and Levesque/Lux and Steiner* [14, 51]
- Learning social behaviors. *Mataric* [53]
- Bayesian learning in negotiation: model others. *Zeng and Sycara* [96]
- Multiagent Q-learning. *Weiss* [93]
- Training other agents' Q-functions (track driving). *Clouse* [13]
- Minimize the need for training. *Potter and Grefenstette* [61]
- Cooperative co-evolution. *Bull et al.* [11]
- Contract nets for electronic commerce. *Sandholm and Lesser* [73]
- Market-based systems. *Huberman and Clearwater* [41]
- Generalized Partial Global Planning (GPGP). *Decker* [21]
- Internal, Social, and Collective (role) commitments. *Castelfranchi* [12]
- Commitment states (potential, pre, and actual) as planning states. *Haddadi* [34]
- Belief/Desire/Intention (BDI) model: OASIS. *Rao and Georgeff* [63]
- BDI commitments only over intentions. *Rao and Georgeff* [63]
- Coalitions. *Zlotkin and Rosenschein/Shehory and Kraus/Sandholm and Lesser* [97, 78, 72]
- Reasoning about truthfulness. *Sandholm and Lesser/ Rosenschein* [74, 65]

Table 7: The issues, techniques, and learning opportunities for communicating multiagent systems as reflected in the literature.

6.3.1 Understanding each other

In all communicating multiagent systems, and particularly in domains that include agents built by different designers, there must be some set language and protocol for the agents to use when interacting. Independent aspects of protocols are information content, message format, and coordination conventions. Among many others, existing language protocols for these three levels are: KIF for content [29], KQML for message format [28], and, more recently, COOL for coordination [8]. There has been a lot of research done on refining these and other communication protocols. MAS designers must carefully consider what features in a communication protocol are needed in a given domain.

6.3.2 Planning communicative acts

With the addition of communication as a capability available to agents, it is possible to consider this capability as an "action" no different from any other. When an agent transmits information to another agent, it has an effect just like any other action would have. Thus within a planning framework, one can define preconditions and effects for communicative acts. When combined with a model of other agents, the effect of a communication act might be to alter an agent's belief about the state of another agent or agents. The theory of communication as action is called *speech acts* [14, 51].

Mataric adds a learning dimension to the idea of speech acts. Starting with the foraging behavior mentioned above [52], the agents can then learn to choose among a set of social behaviors that include broadcasting and listening [53]. Q-learning is extended so that reinforcement can be received for direct rewards or for rewards to other agents.

When using communication as a planning action, the possibility arises of communicating misinformation in order to satisfy a particular goal. For instance, an agent may want another agent to believe that something is true. Rather than actually making it true, the agent might just *say* that it is true. For example, Sandholm and Lesser analyze a framework in which agents are allowed to "decommit" from agreements with other agents by paying a penalty to these other agents [74]. They consider the case in which an agent might not be truthful in its decommitment, hoping that the other agent will decommit first. In such situations, agents must also consider what communications to believe /citeRosenschein94:Rules.

6.3.3 Benevolence vs. competitiveness

Several studies involving competitive agents were described in the heterogeneous non-communicating scenario (see Section 5). In the current scenario, there are many more examples of competitive agents.

Zeng and Sycara study a competitive negotiation scenario in which agents use Bayesian Learning techniques to update models of each other based on bids and counter bids in a negotiation process [96].

Similar to Tan's work on multiagent RL in the pursuit domain [87] is Weiß's work with competing Q-learners. The agents compete with each other to earn the right to control a single system [93]. The highest bidder pays a certain amount to be allowed to act, then receives any reward that results from the action.

Another Q-learning approach, this time with benevolent agents, has been to explore the interesting idea of having one agent teach another agent through communication. Starting with a *trainer* that has moderate expertise in a task, a learner can be rewarded for mimicking the trainer. Furthermore, the trainer can recommend to the learner what action to take in a given situation so as to direct the learner toward a reward state. Eventually, the learner is able to perform the task without any guidance. Clouse studies the effect of different levels of advice in a road-following domain [13]. He concludes that moderate advice improves performance and speeds up learning, while too much advice leads to worse performance because the learner does not experience enough negative examples while training.

While training is a useful concept, some research is driven by the goal of reducing the role of the human trainer. As opposed to the process of *shaping*, in which the system designer develops simple behaviors and slowly builds them into more complex ones, populations appropriately seeded for competitive co-evolution can reduce the amount of designer effort. Potter and Grefenstette illustrate this effect in the domain described above in which two robots compete for a stationary pellet of food [61]. Subpopulations of rules are seeded to be more effective in different situations. Thus specialized subpopulations of rules corresponding to shaped behaviors tend to emerge.

Rather than *competitive* co-evolution Bull et al. build a system system which uses *cooperative* co-evolution [11]. They use GAs to evolve separate communicating agents to control different legs of a quadrapedal robot.

Drawing inspiration from competition in human societies, several researchers have designed systems based on the law of supply and demand. In the *contract nets* framework, agents all have their own goals, are self-interested, and have limited reasoning resources [80]. They bid to accept tasks from other agents and then can either perform the tasks (if they have the proper resources) or subcontract them to still other agents. Agents must pay to contract their tasks out and thus shop around for the lowest bidder. Sandholm and Lesser discuss some of the issues that arise in contract nets [73].

In a similar spirit is an implemented multiagent system that controls air temperature in different rooms of a building [41]. A person can set one's thermostat to any temperature. Then depending on the actual air temperature, the agent for that room tries to "buy" either hot or cold air from another room that has an excess. At the same time, the agent can sell the excess air at the current temperature to other rooms. Modeling the loss of heat in the transfer from one room to another, the agents try to buy and sell at the best possible prices. The market regulates itself to provide equitable usage of a shared resource.

6.3.4 Resource management

In the previous scenario, resource management came up as a problem involving interdependent actions. In the current scenario, agents can also coordinate schedules. Decker's Generalized Partial Global Planning (GPGP) allows several heterogeneous agents to post constraints, or commitments to do a task by some time, to each other's local schedulers and thus coordinate without the aid of any centralized agent [21].

6.3.5 Commitment/decommitment

When agents communicate, they may decide to cooperate on a given task or for a given amount of time. In so doing, they make *commitments* to each other. Committing to another agent involves agreeing to pursue a given goal, possibly in a given manner, regardless of how much it serves one's own interests. Commitments can make systems run much more smoothly by providing a way for agents to "trust" each other, yet it is not obvious how to get self-interested agents to commit to others in a reasonable way. The theory of commitment and decommitment (when the commitment terminates) has consequently drawn considerable attention.

For example, Castelfranchi defines three types of commitment: internal commitment—an agent binds itself to do something, social commitment—an agent commits to another agent, and collective commitment—an agent agrees to fill a certain role [12]. Setting an alarm clock is an example of internal commitment to wake up at a certain time.

Haddadi discusses commitment states as planning states: potential cooperation, pre-commitment, and commitment [34]. Agents can then use means-ends analysis to plan for goals in terms of commitment opportunities. This work is conducted within a model called Belief/Desire/Intention, or BDI.

BDI is a popular technique for modeling other agents. Other agents' domain knowledge (beliefs) and goals (desires) are modeled as well as their "intentions," or goals they are currently trying to achieve and the methods by which they are trying to achieve them. Rao and Georgeff use the BDI model to build a system for air-traffic control, OASIS, which has been implemented for testing (in parallel with human operators who retain full control) at the airport in Sydney, Australia [63]. Each aircraft is represented by a controlling agent which deals with a global sequencing agent. OASIS mixes reactive and deliberative actions in the agents: they can break out of planned sequences when coming across situations that demand immediate reaction. Since agents cannot control their beliefs or desires, they can only make commitments to each other regarding their intentions.

Finally, groups of agents may decide to commit to each other. Rather than the more usual two-agent or all-agent commitment scenarios, Zlotkin and Rosenschein study situations in which agents may want to form coalitions [97]. Since this work is conducted in a game theory framework, agents consider the utility of joining a coalition in which they are bound to try to advance the utility of other members in exchange for reciprocal consideration. Shehory and Kraus present a a distributed algorithm for task allocation when coalitions are either needed to perform tasks or more efficient that single agents [78]. Sandholm and Lesser use a vehicle routing domain to illustrate a method by which agents can form valuable coalitions when it is intractable to discover the optimal coalitions [72].

6.4 Further Learning Opportunities

Once again, there are many possible ways in the current scenario to enhance MAS with ML techniques. Within this heterogeneous communicating multiagent scenario there is a clear need to pre-define a language and communication protocol for use by the agents. However, an interesting alternative would be to allow the agents to learn for themselves what to communicate and how to interpret it. For example, an agent might be given a small language of utterances and a small set of meanings, but no mapping between the two. Agents would then have to learn both what to say and how to interpret what they hear. A possible result would be more efficient communications: they would need to be understandable only by the agents rather than by both agents and humans.

When considering communications as speech acts, agents could be allowed to learn the effects of speech on the global dynamics of the system. In domains with low bandwidth or large time delays associated with communication, the utility of communicating at a given moment might be learned. In addition, if allowed to learn to communicate, agents are more likely to avoid being reliably conned by untruthfulness in communication: when another agent says something that turns out not to be true, it will not be believed so readily in the future.

Finally, commitment—the act of taking on another agent's goals—has both benefits and disadvantages. System builders may want to allow their agents to learn when to commit to others. The learning opportunities in this scenario are summarized in Table 7.

7 Robotic Soccer

Several multiagent domains have been mentioned throughout the course of this survey, including design, planning, entertainment, games, air-traffic control, air combat, personal assistants, load-balancing, and robotic leg control. In this section a single domain which embodies most multiagent issues is presented.

Robotic soccer is a particularly good domain for studying MAS. Originated by Alan Mackworth [70], it has been gaining popularity in recent years, with several international competitions taking place [43, 44, 33]. It is also the subject of an official IJCAI-97 Challenge [45]. It can be used to evaluate different MAS techniques in a direct manner: teams implemented with different techniques can play against each other.

Although the pursuit domain serves us well for purposes of illustration, robotic soccer is much more complex and interesting as a general testbed for MAS. Even with many predators and several prey, the pursuit domain is not complex enough to simulate the real world. Although robotic soccer is a game, most real-world complexities are retained. A key aspect of soccer's complexity is the need for agents not only to control themselves, but also to control the ball which is a passive part of the environment.

7.1 Overview

Robotic soccer can be played either with real robots or in a simulator. Although more costly and time consuming to develop, a number of groups have developed real robotic systems. The first robotic soccer system was the Dynamo system [70]. Sahota et al. built a 1 vs. 1 version of the game. Asada et al. have used vision-based RL with their soccer playing robots [5]. Veloso et al. discuss some of the robotic issues involved in building robotic soccer players [3, 89].

Some robotic issues can only be studied in the real-world instantiation, but there are also many issues that can be studied in simulation. A particularly good simulator for this purpose is the "soccerserver" developed by Noda [56] and pictured in Figure 12. This simulator is realistic in many ways: the players' vision is limited; the players can communicate by posting to a blackboard that is visible to all players; all players are controlled by separate processes; each player has 10 teammates and 11 opponents; each player has limited stamina; actions and sensors are noisy; and play occurs in real time. The simulator provides a domain and supports users who wish to build their own agents. Furthermore, teams of agents can be evaluated by playing against each other, or perhaps against standard teams. The simulator was successfully used for a competition among twenty-nine teams from around the world in 1997 [44]. Thus robotic soccer satisfies Decker's criteria for DAI testbeds [18].

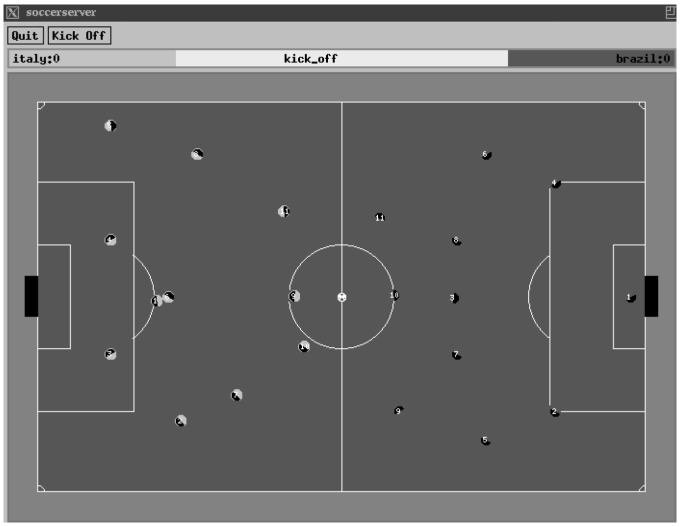

Figure 12: The soccerserver system

7.2 MAS in Robotic Soccer

The main goal of any testbed is to facilitate the trial and evaluation of ideas that have promise in the real world. A wide variety of MAS issues can be studied in simulated robotic soccer. In fact, all of the seventeen MAS issues listed in Table 2 can be feasibly studied in the soccer simulator. The advantages of robotic soccer as a testbed for MAS are summarized in Table 8.

Table 8: Advantages of (simulated) robotic soccer as a MAS testbed

• Complex enough to be realistic	• Direct comparisons possible
• Easily accessible	• Good multiagent ML opportunities
• Embodies most MAS issues	

Homogeneous non-communicating MAS can be studied in robotic soccer by fixing the behavior of the opposing team and populating the team being studied with identical, mute players. To keep within the homogeneous agent scenario, the opponents must not be modeled as agents. In this context, the players can be reactive or deliberative to any degree. The extremely reactive agent might simply look for the ball and move straight at it, shooting whenever possible. At this extreme, the players may or may not have any knowledge that they are part of a team. On the other hand, players might model each other, thus enabling deliberative reasoning about whether to approach the ball or whether to move to a different part of the field in order to defend or to receive a pass. With players modeling each other, they may also reason about how to affect each other's behaviors in this inherently dynamic environment. Finally it is possible to study the relative merits of local and global perspectives on the world. Robots can be given global views with the help of an overhead camera, and the soccerserver comes equipped with a coach mode that permits global views. However, robotic soccer is usually approached as a problem requiring local sensing.

Robotic soccer is also useful for studying the issues associated with heterogeneous non-communicating agents. Since each player has several teammates with the same global goal and several opponents with the diametrically opposed goal, each player is both benevolent and competitive at the same time. This possibility for combination of collaborative and adversarial reasoning is a major feature of the domain. When trying to collaborate, players' actions are usually interdependent: to execute a successful pass, both the passer and the receiver must execute the appropriate actions. Thus modeling each other for the purpose of coordination is helpful. Social conventions, such as programmed notions of when a given agent will pass or which agents should play defense, can also help coordination. Since communication is still not allowed, the players must have a reliable method for filling the different team roles needed on a soccer team (defense, offense, goalie). Ideally, the players are able to switch roles during the course of a game when appropriate. Finally, if the teams are learning during the course of a single game or over several games, all the issues of evolving agents, including the "arms race" possibility and the credit/blame problem, arise.

Robotic soccer is perhaps best suited for the study of the most complex multiagent scenario: heterogeneous communicating agents. Since the agents can indeed communicate, the full potential of the domain is realized in this scenario. With players posting messages to the blackboard, they must have a language in order to understand each other. Protocols are also needed for commitment to team plays: the passer and receiver in a pass play must both agree to execute the pass. For more complex team plays, several players may need to commit to participate. But then the issue arises of how single-mindedly they must adhere to the committed play: when may they react to more pressing situations and ignore the commitment? For any team play, including a simple pass, timing is very important in such a real-time scenario. Thus, players must coordinate their actions very carefully. Finally, speech acts are particularly interesting in the environment that is both collaborative and adversarial. If the opponents can understand the same language, a planned utterance can affect the knowledge of both teammates and opponents. The utility of communication must be carefully considered and the possibility of lying in order to fool the opponent arises. Therefore, planned communicative acts, along with most of the other issues from Table 2, turn up in robotic soccer.

In terms of the reasons to use MAS presented in Table 1, robotic soccer systems usually require separate agents for controlling the separate players, and they can benefit from the parallelism, robustness, and simpler programming of MAS. Systems whose players have onboard sensors are necessarily multiagent, since no single agent has access to all of the players' sensory inputs. Some competitions also stipulate in their rules that the robots must be controlled by separate agents. At the very least, the two teams must be controlled by separate agents. Even teams that could theoretically be controlled by a single agent stand to gain by using MAS. By processing the sensory inputs of the different players separately, multiple agents can control their players in parallel, perhaps contending with different tasks on the field. One player might be in position to defend its goal, while another is preparing an offensive attack. These players need not be controlled by the same agent: they can go about their tasks in parallel. Furthermore, if any of the agents fails for some reason (as often happens in real robotic systems), the other agents can attempt to compensate and continue playing. Finally, it is empirically easier to program a single agent per player than it is to control an entire team centrally.

7.3 Machine Learning in Robotic Soccer

As well as addressing most of the issues inherent in MAS, robotic soccer is a great domain for multiagent Machine Learning. In another soccer simulator, Stone and Veloso use Memory-based Learning to allow a player to learn when to shoot and when to pass the ball [82]. They then use Neural Networks to teach a player to shoot a moving ball into the goal [83]. They use similar techniques in the soccerserver system as well, extending the learned behavior as a part of a hierarchical learning system [84]. Matsubara et al. also use a Neural Network to allow a player to learn when to shoot and when to pass in the soccerserver system [54]. Uchibe et al. have successfully combined RL modules for shooting and for avoiding opponents using real robots [88].

Once low-level behaviors have been developed, the opportunity to use ML techniques at the strategy level is particularly exciting. For example, Balch uses a behavioral diversity measure to encourage role learning in a RL framework, finding that providing a uniform reinforcement to the entire team is more effective than providing local reinforcements to individual players [6]. Luke et al. use genetic programming to evolve cooperative behaviors within a team of players [50].

8 Conclusion

This survey is presented as a description of the field of MAS. It is designed to serve both as an introduction for people unfamiliar with the field and as an organizational framework for system designers. This framework is presented as a series of three increasingly complex and powerful scenarios. The simplest systems are those with homogeneous non-communicating agents. The second scenario involves heterogeneous non-communicating agents. Finally, the general MAS scenario involves communicating agents with any degree of heterogeneity. Single-agent systems are presented as the most extreme version of this final, most complex scenario, where control is centralized in one agent and the others act as remote slaves.

Each multiagent scenario introduces new issues and complications. Although MAS is a new field, several techniques and systems already address these issues. After summarizing a wide range of such existing work, useful future directions are presented. Throughout the survey, Machine Learning approaches are emphasized.

Although each domain requires a different approach, from a research perspective the ideal domain embodies as many issues as possible. Robotic soccer is presented here as a useful domain for the study of MAS. Systems with a wide variety of agent heterogeneity and communication abilities can be studied. In addition, collaborative and adversarial issues can be combined in a real-time situation. With the aid of research in such complex domains, the field of MAS should continue to advance and to spread in popularity among designers of real systems.

MAS is an active field with many open issues. Continuing research is presented at dedicated conferences and workshops such as the International Conference on Multi-Agent Systems [95, 2, 1]. MAS work also appears in many of the DAI conferences and workshops [22, 94]. This survey provides a framework within which the reader can situate both existing and future work.

Acknowledgements

We would like to thank Keith Decker, Rala Stone, Russell Stone, Astro Teller, and the anonymous reviewers for their helpful comments and suggestions.

This research is sponsored by the Wright Laboratory, Aeronautical Systems Center, Air Force Materiel Command, USAF, and the Advanced Research Projects Agency (ARPA) under grant number F33615-93-1-1330. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Wright Laboratory or the U. S. Government.

References

[1] AAAI. *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, Menlo Park,CA, June 1995. AAAI Press. Victor Lessor–General Chair.

[2] AAAI. *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01. Sandip Sen–Chair.

[3] Sorin Achim, Peter Stone, and Manuela Veloso. Building a dedicated robotic soccer system. In *Proceedings of the IROS-96 Workshop on RoboCup*, November 1996.

[4] Neeraj Arora and Sandip Sen. Resolving social dilemmas using genetic algorithms. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 1–5, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[5] M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, and K. Hosoda. Coordination of multiple behaviors acquired by vision-based reinforcement learning. In *Proc. of IEEE/RSJ/GI International Conference on Intelligent Robots and Systems 1994 (IROS '94)*, pages 917–924, 1994.

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

[7] Tucker Balch and Ronald C. Arkin. Motor schema-based formation control for multiagent robot teams. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 10–16, Menlo Park, California, June 1995. AAAI Press.

[8] Mihai Barbuceanu and Mark S. Fox. Cool: A language for describing coordination in multi agent systems. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 17–24, Menlo Park, California, June 1995. AAAI Press.

[9] 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 Computing Services, Seattle, Washington, July 1986.

[10] Alan H. Bond and Les Gasser. An analysis of problems and research in DAI. In Alan H. Bond and Les Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 3–35. Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[11] L. Bull, T.C. Fogarty, and M. Snaith. Evolution in multi-agent systems: Evolving communicating classifier systems for gait in a quadrupedal robot. In Stephanie Forrest, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 382–388, San Mateo,CA, July 1995. Morgan Kaufman.

[12] Cristiano Castelfranchi. Commitments: From individual intentions to groups and organizations. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 41–48, Menlo Park, California, June 1995. AAAI Press.

[13] Jeffery A. Clouse. Learning from an automated training agent. In Gerhard Weißand Sandip Sen, editors, *Adaptation and Learning in Multiagent Systems*. Springer Verlag, Berlin, 1996.

[14] Philip R. Cohen and Hector J. Levesque. Communicative actions for artificial agents. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 65–72, Menlo Park, California, June 1995. AAAI Press.

[15] Kerstin Dautenhahn. Getting to know each other—artificial social intelligence for autonomous robots. *Robotics and Autonomous Systems*, 16:333–356, 1995.

[16] Keith S. Decker. Distributed problem solving: A survey. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(5):729–740, September 1987.

[17] Keith S. Decker. *Environment Centered Analysis and Design of Coordination Mechanisms*. PhD thesis, University of Massachusetts, 1995.

[18] Keith S. Decker. Distributed artificial intelligence testbeds. In G. M. P. O'Hare and N. R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, pages 119–138. Wiley Interscience, 1996.

[19] Keith S. Decker. Personal correspondence, May 1996.

[20] Keith S. Decker. Task environment centered simulation. In M. Prietula, K. Carley, and L. Gasser, editors, *Simulating Organizations: Computational Models of Institutions and Groups*. AAAI Press/MIT Press, 1996.

[21] Keith S. Decker and Victor R. Lesser. Designing a family of coordination algorithms. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 73–80, Menlo Park, California, June 1995. AAAI Press.

[22] *Proceedings of the 10th International Workshop on Distributed Artificial Intelligence*, Bandera, Texas, October 1990.

[23] Edmund H. Durfee. What your computer really needs to know, you learned in kindergarten. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, Philadelphia, PA, 1992. Morgan Kaufman. Invited Talk.

[24] Edmund H. Durfee. Blissful ignorance: Knowing just enough to coordinate well. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 406–413, Menlo Park, California, June 1995. AAAI Press.

[25] Edmund H. Durfee, Victor R. Lesser, and Daniel D. Corkill. Trends in cooperative distributed problem solving. *IEEE Transactions on Knowledge and Data Engineering*, 1(1):63–83, March 1989.

[26] Maier Fenster, Sarit Kraus, and Jeffrey S. Rosenschein. Coordination without communication: Experimental validation of focal point techniques. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 102–108, Menlo Park, California, June 1995. AAAI Press.

[27] Innes A. Ferguson and Grigoris J. Karakoulas. Multiagent learning and adaptation in an information filtering market. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 28–32, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[28] Tim Finin, Don McKay, Rich Fritzson, and Robin McEntire. Kqml: An information and knowledge exchange protocol. In Kazuhiro Fuchi and Toshio Yokoi, editors, *Knowledge Building and Knowledge Sharing*. Ohmsha and IOS Press, 1994.

[29] M. R. Genesereth and R. E. Fikes. Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992.

[30] Natalie S. Glance and Tad Hogg. Dilemmas in computational societies. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 117–124, Menlo Park, California, June 1995. AAAI Press.

[31] Claudia Goldman and Jeffrey Rosenschein. Emergent coordination through the use of cooperative state-changing rules. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 408–413, Philadelphia, PA, 1994. Morgan Kaufman.

[32] John Grefenstette and Robert Daley. Methods for competitive and cooperative co-evolution. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 45–50, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[33] Fred Gruau. Announcement for participation of the first autonomous robotics football tournament, 1997. Accessible from http://www.dcs.qmw.ac.uk/research/ai/robot/football/FirstARFTannouncement.html.

[34] Afsaneh Haddadi. Towards a pragmatic theory of interactions. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 133–139, Menlo Park, California, June 1995. AAAI Press.

[35] Barbara Hayes-Roth, Lee Brownston, and Robert van Gent. Multiagent collaboration in directed improvisation. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 148–154, Menlo Park, California, June 1995. AAAI Press.

[36] Thomas Haynes, Kit Lau, and Sandip Sen. Learning cases to compliment rules for conflict resolution in multiagent systems. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 51–56, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[37] Thomas Haynes and Sandip Sen. Evolving behavioral strategies in predators and prey. In Gerhard Weißand Sandip Sen, editors, *Adaptation and Learning in Multiagent Systems*, pages 113–126. Springer Verlag, Berlin, 1996.

[38] Thomas Haynes, Roger Wainwright, Sandip Sen, and Dale Schoenefeld. Strongly typed genetic programming in evolving cooperation strategies. In Stephanie Forrest, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 271–278, San Mateo, CA, July 1995. Morgan Kaufman.

[39] O.E. Holland. Multiagent systems: Lessons from social insects and collective robotics. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 57–62, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[40] Marcus J. Huber and Edmund H. Durfee. Deciding when to commit to action during observation-based coordination. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 163–170, Menlo Park, California, June 1995. AAAI Press.

[41] Bernardo Huberman and Scott H. Clearwater. A multi-agent system for controlling building environments. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 171–176, Menlo Park, California, June 1995. AAAI Press.

[42] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, May 1996.

[43] Jong-Hwan Kim, editor. *Proceedings of the Micro-Robot World Cup Soccer Tournament*, Taejon, Korea, November 1996.

[44] H. Kitano, Y. Kuniyoshi, I. Noda, M. Asada, H. Matsubara, and E. Osawa. Robocup: A challenge problem for AI. *AI Magazine*, 18(1):73–85, Spring 1997.

[45] Hiroaki Kitano, Milind Tambe, Peter Stone, Manuela Veloso, Silvia Coradeschi, Eiichi Osawa, Hitoshi Matsubara, Itsuki Noda, and Minoru Asada. The robocup synthetic agent challenge 97. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, San Francisco, CA, 1997. Morgan Kaufman.

[46] Richard E. Korf. A simple solution to pursuit games. In *Working Papers of the 11th International Workshop on Distributed Artificial Intelligence*, pages 183–194, February 1992.

[47] Victor R. Lesser. Multiagent systems: An emerging subdiscipline of AI. *ACM Computing Surveys*, 27(3):340–342, September 1995.

[48] Ran Levy and Jeffrey S. Rosenschein. A game theoretic approach to the pursuit problem. In *Working Papers of the 11th International Workshop on Distributed Artificial Intelligence*, pages 195–213, February 1992.

[49] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163, San Mateo, CA, 1994. Morgan Kaufman.

[50] Sean Luke, Charles Hohn, Jonathan Farris, Gary Jackson, and James Hendler. Co-evolving soccer softbot team coordination with genetic programming. In *Proceedings of the First International Workshop on RoboCup*, pages 115–118, Nagoya,Japan, August 1997.

[51] Andreas Lux and Donald Steiner. Understanding cooperation: an agent's perspective. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 261–268, Menlo Park, California, June 1995. AAAI Press.

[52] Maja J. Mataric. Interaction and intelligent behavior. MIT EECS PhD Thesis AITR-1495, MIT AI Lab, August 1994.

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

[54] Hitoshi Matsubara, Itsuki Noda, and Kazuo Hiraki. Learning of cooperative actions in multi-agent systems: a case study of pass play in soccer. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 63–67, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[55] Yishay Mor and Jeffrey Rosenschein. Time and the prisoner's dilemma. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 276–282, Menlo Park, California, June 1995. AAAI Press.

[56] Itsuki Noda. Soccer server : a simulator of robocup. In *Proceedings of AI symposium '95*, pages 29–34. Japanese Society for Artificial Intelligence, December 1995.

[57] Ei-Ichi Osawa. A metalevel coordination strategy for reactive cooperative planning. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 297–303, Menlo Park, California, June 1995. AAAI Press.

[58] H. Van Dyke Parunak. Applications of distributed artificial intelligence in industry. In G. M. P. O'Hare and N. R. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, pages 139–164. Wiley Interscience, 1996.

[59] Yongyuth Permpoontanalarp. Generalised proof-theory for multi-agent autoepistemic reasoning. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 304–311, Menlo Park, California, June 1995. AAAI Press.

[60] Dean A. Pormerleau. *Neural Network Perception for Mobile Robot Guidance*. Kluwer Academic Publishers, 1993.

[61] Mitchell A. Potter, Kenneth A. De Jong, and John J. Grefenstette. A coevolutionary approach to learning sequential decision rules. In Stephanie Forrest, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 366–372, San Mateo,CA, July 1995. Morgan Kaufman.

[62] M V Nagendra Prasad, Victor R. Lesser, and Susan E. Lander. Learning organizational roles in a heterogeneous multi-agent system. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 72–77, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[63] Anand S. Rao and Michael P. Georgeff. Bdi agents: From theory to practice. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 312–319, Menlo Park, California, June 1995. AAAI Press.

[64] Matt Ridley. *The Origins of Virtue : Human Instincts and the Evolution of Cooperation*. Viking Press, April 1997.

[65] Jeffrey S Rosenschein and Gilad Zlotkin. *Rules of Encounter*. MIT Press, 1994.

[66] Christopher D. Rosin and Richard K. Belew. Methods for competitive co-evolution: Finding opponents worth beating. In Stephanie Forrest, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 373–380, San Mateo,CA, July 1995. Morgan Kaufman.

[67] Shounak Roychowdhury, Neeraj Arora, and Sandip Sen. Effects of local information on group behavior. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 78–83, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[68] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[69] Michael K. Sahota. Reactive deliberation: An architecture for real-time intelligent control in dynamic environments. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1303–1308, 1994.

[70] Michael K. Sahota, Alan K. Mackworth, Rod A. Barman, and Stewart J. Kingdon. Real-time control of soccer-playing robots using off-board vision: the dynamite testbed. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 3690–3663, 1995.

[71] J. Alfredo Sanchez, Flavio S. Azevedo, and John J. Leggett. Paragente: Exploring the issues in agent-based user interfaces. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 320–327, Menlo Park, California, June 1995. AAAI Press.

[72] Tuomas Sandholm and Victor Lesser. Coalition formation among bounded rational agents. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 662–669, Los Angeles, CA, 1995. Morgan Kaufman.

[73] Tuomas Sandholm and Victor Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 328–335, Menlo Park, California, June 1995. AAAI Press.

[74] Tuomas Sandholm and Victor Lesser. Advantages of a leveled commitment contracting protocol. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 126–133, Menlo Park,California, 1996. AAAI Press.

[75] Tuomas W. Sandholm and Robert H. Crites. On multiagent q-learning in a semi-competitive domain. In Gerhard Weißand Sandip Sen, editors, *Adaptation and Learning in Multiagent Systems*. Springer Verlag, Berlin, 1996.

[76] Andrea Schaerf, Yoav Shoham, and Moshe Tennenholtz. Adaptive load balancing: A study in multi-agent learning. *Journal of Artificial Intelligence Research*, 2:475–500, 1995.

[77] Jurgen Schmidhuber. A general method for multi-agent reinforcement learning in unrestricted environments. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 84–87, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[78] Onn Shehory and Sarit Kraus. Task allocation via coalition formation among autonomous agents. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 655–661, Los Angeles, CA, 1995. Morgan Kaufman.

[79] Jaime Simao Sichman and Yves Demazeau. Exploiting social reasoning to deal with agency level inconistency. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 352–359, Menlo Park, California, June 1995. AAAI Press.

[80] Reid G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, December 1980.

[81] Larry M. Stephens and Matthias B. Merx. The effect of agent control strategy on the performance of a dai pursuit problem. In *Proceedings of the 10th International Workshop on Distributed Artificial Intelligence*, Bandera, Texas, October 1990.

[82] Peter Stone and Manuela Veloso. Beating a defender in robotic soccer: Memory-based learning of a continuous function. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 896–902, Cambridge, MA, 1996. MIT press.

[83] Peter Stone and Manuela Veloso. Towards collaborative and adversarial learning: A case study in robotic soccer. *To appear in International Journal of Human-Computer Systems (IJHCS)*, 1997.

[84] Peter Stone and Manuela Veloso. A layered approach to learning client behaviors in the robocup soccer server. *To appear in Applied AI Journal*, 1998.

[85] Milind Tambe. Recursive agent and agent-group tracking in a real-time , dynamic environment. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 368–375, Menlo Park, California, June 1995. AAAI Press.

[86] Milind Tambe. Tracking dynamic team activity. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Menlo Park, California, 1996. AAAI Press.

[87] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 330–337, 1993.

[88] Eiji Uchibe, Minoru Asada, and Koh Hosoda. Behavior coordination for a mobile robot using modular reinforcement learning. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems 1996 (IROS '96)*, pages 1329–1336, 1996.

[89] Manuela Veloso, Peter Stone, Kwun Han, and Sorin Achim. Cmunited: A team of robotic soccer agents collaborating in an adversarial environment. In *Proceedings of the First International Workshop on RoboCup*, Nagoya,Japan, August 1997.

[90] Jose M. Vidal and Edmund H. Durfee. Recursive agent modeling using limited rationality. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 376–383, Menlo Park, California, June 1995. AAAI Press.

[91] Adam Walker and Michael Wooldridge. Understanding the emergence of conventions in multi-agent systems. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 384–389, Menlo Park, California, June 1995. AAAI Press.

[92] Xuemei Wang. Planning while learning operators. In *Proceedings of the Third International Conference on AI Planning Systems*, May 1996.

[93] Gerhard Weiß. Distributed reinforcement learning. *Robotics and Autonomous Systems*, 15:135–142, 1995.

[94] Gerhard Weiß. Ecai-96 workshop on learning in distributed artificial intelligence. Call For Papers, 1996.

[95] Gerhard Weiß and Sandip Sen, editors. *Adaptation and Learning in Multiagent Systems*. Springer Verlag, Berlin, 1996.

[96] Dajun Zeng and Katia Sycara. Bayesian learning in negotiation. In *Adaptation, Coevolution and Learning in Multiagent Systems: Papers from the 1996 AAAI Spring Symposium*, pages 99–104, Menlo Park,CA, March 1996. AAAI Press. AAAI Technical Report SS-96-01.

[97] Gilad Zlotkin and Jeffrey S. Rosenschein. Coalition, cryptography, and stability: Mechanisms for coalition formation in task oriented domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 432–437, Menlo Park, California, August 1994. AAAI Press.