AI Magazine Volume 19 Number 2 (1998) (© AAAI)

Multiagent Systems

Katia P. Sycara

■ Agent-based systems technology has generated lots of excitement in recent years because of its promise as a new paradigm for conceptualizing, designing, and implementing software systems. This promise is particularly attractive for creating software that operates in environments that are distributed and open, such as the internet. Currently, the great majority of agent-based systems consist of a single agent. However, as the technology matures and addresses increasingly complex applications, the need for systems that consist of multiple agents that communicate in a peer-to-peer fashion is becoming apparent. Central to the design and effective operation of such multiagent systems (MASs) are a core set of issues and research questions that have been studied over the years by the distributed AI community. In this article, I present some of the critical notions in MASs and the research work that has addressed them. I organize these notions around the concept of problem-solving coherence, which I believe is one of the most critical overall characteristics that an MAS should exhibit.

Most researchers in AI to date have dealt with developing theories, techniques, and systems to study and understand the behavior and reasoning properties of a single cognitive entity. AI has matured, and it endeavors to attack more complex, realistic, and large-scale problems. Such problems are beyond the capabilities of an individual agent. The capacity of an intelligent agent is limited by its knowledge, its computing resources, and its perspective. This bounded rationality (Simon 1957) is one of the underlying reasons for creating problem-solving organizations. The most powerful tools for handling complexity are modularity and abstraction. Multiagent systems (MASs) offer modularity. If a problem domain is particularly complex, large, or unpredictable, then the only way it can reasonably be addressed is to develop a number of functionally specific and (nearly) modular components (agents) that are specialized at solving a particular problem aspect. This decomposition allows each agent to use the most appropriate paradigm for solving its particular problem. When interdependent problems arise, the agents in the system must coordinate with one another to ensure that interdependencies are properly managed.

Furthermore, real problems involve distributed, open systems (Hewitt 1986). An *open system* is one in which the structure of the system itself is capable of dynamically changing. The characteristics of such a system are that its components are not known in advance; can change over time; and can consist of highly heterogeneous agents implemented by different people, at different times, with different software tools and techniques. Perhaps the best-known example of a highly open software environment is the internet. The internet can be viewed as a large, distributed information resource, with nodes on the network designed and implemented by different organizations and individuals. In an open environment, information sources, communication links, and agents could appear and disappear unexpectedly. Currently, agents on the internet mostly perform information retrieval and filtering. The next generation of agent technology will perform information gathering in context and sophisticated reasoning in support of user problem-solving tasks. These capabilities require that agents be able to interoperate and coordinate with each other in peer-to-peer interactions. In addition, these capabilities will allow agents to increase the problem-solving scope of single agents. Such functions will require techniques based on negotiation or cooperation, which lie firmly in the domain of MASs (Jennings, Sycara, and Wooldridge 1998;

O'Hare and Jennings 1996; Bond and Gasser 1988).

It is becoming increasingly clear that to be successful, increased research resources and attention should be given to systems consisting of not one but multiple agents. The distributed AI (DAI) community that started forming in the early 1980s and was tiny compared to mainstream, single-agent AI is rapidly increasing. The growth of the MAS field is indisputable.

Research in MASs is concerned with the study, behavior, and construction of a collection of possibly preexisting autonomous agents that interact with each other and their environments. Study of such systems goes beyond the study of individual intelligence to consider, in addition, problem solving that has social components. An *MAS* can be defined as a loosely coupled network of problem solvers that interact to solve problems that are beyond the individual capabilities or knowledge of each problem solver (Durfee and Lesser 1989). These problem solvers, often called *agents*, are autonomous and can be heterogeneous in nature.

The characteristics of MASs are that (1) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (2) there is no system global control; (3) data are decentralized; and (4) computation is asynchronous.

The characteristics of MASs are that (1) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (2) there is no system global control; (3) data are decentralized; and (4) computation is asynchronous. The motivations for the increasing interest in MAS research include the ability of MASs to do the following:

First is to solve problems that are too large for a centralized agent to solve because of resource limitations or the sheer risk of having one centralized system that could be a performance bottleneck or could fail at critical times.

Second is to allow for the interconnection and interoperation of multiple existing legacy systems. To keep pace with changing business needs, legacy systems must periodically be updated. Completely rewriting such software tends to be prohibitively expensive and is often simply impossible. Therefore, in the short to medium term, the only way that such legacy systems can remain useful is to incorporate them into a wider cooperating agent community in which they can be exploited by other pieces of software. Incorporating legacy systems into an agent society can be done, for example, by building an agent wrapper around the software to enable it to interoperate with other systems (Genesereth and Ketchpel 1994).

Third is to provide solutions to problems that can naturally be regarded as a society of autonomous interacting components-agents. For example, in meeting scheduling, a scheduling agent that manages the calendar of its user can be regarded as autonomous and interacting with other similar agents that manage calendars of different users (Garrido and Sycara 1996; Dent et al. 1992). Such agents also can be customized to reflect the preferences and constraints of their users. Other examples include air-traffic control (Kinny et al. 1992; Cammarata, McArthur, and Steeb 1983) and multiagent bargaining for buying and selling goods on the internet.

Fourth is to provide solutions that efficiently use information sources that are spatially distributed. Examples of such domains include sensor networks (Corkill and Lesser 1983), seismic monitoring (Mason and Johnson 1989), and information gathering from the internet (Sycara et al. 1996).

Fifth is to provide solutions in situations where expertise is distributed. Examples of such problems include concurrent engineering (Lewis and Sycara 1993), health care, and manufacturing.

Sixth is to enhance performance along the dimensions of (1) *computational efficiency* because concurrency of computation is exploited (as long as communication is kept minimal, for example, by transmitting high-level information and results rather than low-level data); (2) *reliability*, that is, graceful recovery of component failures, because agents with redundant capabilities or appropriate interagent coordination are found dynamically (for example, taking up responsibilities of agents that fail); (3) *extensibility* because the number and the capabilities of agents working on a problem can be altered; (4) *robustness*, the system's ability to tolerate uncertainty, because suitable information is exchanged among agents; (5) *maintainability* because a system composed of multiple components-agents is easier to maintain because of its modularity; (6) *responsiveness* because modularity can handle anomalies locally, not propagate them to the whole system; (7) *flexibility* because agents with different abilities can adaptively organize to solve the current problem; and (8) *reuse* because functionally specific agents can be reused in different agent teams to solve different problems.

MASs are now a research reality and are rapidly having a critical presence in many human-computer environments. My purpose in this article is not to provide a detailed review of the field; I leave this task to others (see, for example, Huhns and Singh [1997], O'Hare and Jennings [1996], Wooldridge and Jennings [1995], Chaib-draa et al. [1992], and Bond and

Gasser [1988] for surveys). Rather than present an in-depth analysis and critique of the field, I instead briefly discuss some key topics and indicate how they are interrelated. Where appropriate, references to more detailed treatments are provided.

Multiagent System Issues and Challenges

Although MASs provide many potential advantages, they also present many difficult challenges. Here, I present problems inherent in the design and implementation of MASs. The list of challenges includes problems first posed in Bond and Gasser (1988), but I have added some:

First, how do we formulate, describe, decompose, and allocate problems and synthesize results among a group of intelligent agents?

Second, how do we enable agents to communicate and interact? What communication languages and protocols do we use? How can heterogeneous agents interoperate? What and when can they communicate? How can we find useful agents in an open environment?

Third, how do we ensure that agents act coherently in making decisions or taking action, accommodating the nonlocal effects of local decisions and avoiding harmful interactions? How do we ensure the MAS does not become resource bounded? How do we avoid unstable system behavior?

Fourth, how do we enable individual agents to represent and reason about the actions, plans, and knowledge of other agents to coordinate with them; how do we reason about the state of their coordinated process (for example, initiation and completion)?

Fifth, how do we recognize and reconcile disparate viewpoints and conflicting intentions among a collection of agents trying to coordinate their actions?

Sixth, how do we engineer and constrain practical DAI systems? How do we design technology platforms and development methodologies for MASs?

Solutions to these problems are intertwined (Gasser 1991). For example, different modeling schemes of an individual agent can constrain the range of effective coordination regimes; different procedures for communication and interaction have implications for behavioral coherence. Different problem and task decompositions can yield different interactions. It is arguable whether one can find a unique most important dimension along which a treatment of MASs can cogently be organized. Here, I attempt to use the dimension of effective overall problem-solving coherence of an MAS as the organizing theme.

Ensuring that an MAS exhibits coherent collective behavior while it avoids unpredictable or harmful behavior (for example, chaos, oscillation) is indeed a major challenge: By its very nature, an MAS lacks global perspective, global control, or global data. Coherence is a global (or regional) property of the MAS that could be measured by the efficiency, quality, and consistency of a global solution (system behavior) as well as the ability of the system to degrade gracefully in the presence of local failures. Several methods for increasing coherence have been studied. These methods, along with issues of single-agent structuring in an MAS, cover the topics I want to survey here.

Individual Agent Reasoning

Sophisticated individual agent reasoning can increase MAS coherence because each individual agent can reason about nonlocal effects of local actions, form expectations of the behavior of others, or explain and possibly repair conflicts and harmful interactions. Numerous works in AI research try to formalize a logical axiomatization for rational agents (see Wooldridge and Jennings [1995]) for a survey). This axiomatization is accomplished by formalizing a model for agent behavior in terms of beliefs, desires, goals, and so on. These works are known as belief-desire-intention (BDI) systems (see Rao and Georgeff [1991] and Shoham [1993]). An agent that has a BDI-type architecture has also been called *deliberative*.

In my own work on the RETSINA multiagent infrastructure, agents coordinate to gather information in the context of user problem-solving tasks. Each RETSINA agent is a BDI-type agent that integrates planning, scheduling, execution, information gathering, and coordination with other agents (Decker, Pannu, et al. 1997; Sycara et al. 1996). Each agent has a sophisticated reasoning architecture that consists of different modules that operate asynchronously.

The *planning module* takes as input a set of goals and produces a plan that satisfies the goals. The agent planning process is based on a hierarchical task network (HTN) planning formalism. It takes as input the agent's current set of goals, the current set of task structures, and a library of task-reduction schemas. A *task-reduction schema* presents a way of carrying out a task by specifying a set of subtasks-actions and describing the information-flow relationships between them. (See Williamson, Decker, and Sycara [1996]). The *communication and coordination module* accepts and interprets messages

Sophisticated individual agent reasoning can increase MAS coherence because each individual agent can reason about nonlocal effects of local actions, form expectations of the behavior of others, or explain and possibly repair conflicts and harmful interactions.

from other agents in KQML. Messages can contain requests for services. These requests become goals of the recipient agent. The *scheduling module* schedules each of the plan steps. The agent scheduling process takes as input, in general, the agent's current set of plan instances and, in particular, the set of all executable actions and decides which action, if any, is to be executed next. This action is then identified as a fixed intention until it is actually carried out (by the execution component). Agent-reactivity considerations are handled by the execution-monitoring process. Execution monitoring takes as input the agent's next intended action and prepares, monitors, and completes its execution. The *execution monitor* prepares an action for execution by setting up a context (including the results of previous actions) for the action. It monitors the action by optionally providing the associated computation-limited resources—for example, the action might be allowed only a certain amount of time, and if the action does not complete before the time is up, the computation is interrupted, and the action is marked as having failed. Failed actions are handled by the *exception-handling process.* The agent has a domain-independent library of plan fragments (task structures) that are indexed by goals as well as a domain-specific library of plan fragments that can be retrieved and incrementally instantiated according to the current input parameters.

Reactive agents have also been developed. Reactive agents have their roots in Brooks's (1991) criticism of deliberative agents and his assertions that (1) intelligence is the product of the interaction of an agent and its environment and (2) intelligent behavior emerges from the interaction of various simpler behaviors organized in a layered way through a master-slave relationship of inhibition (subsumption architecture). A different reactive architecture is based on considering the behavior of an agent as the result of competing entities trying to get control over the actions of the agent. This idea has its root in the society of the mind (Minsky 1986) and has been employed in different ways by Maes (1990), Travers (1988), and Drogul and Ferber (Ferber 1996; Drogul and Ferber 1992). An *agent* is defined as a set of conflicting tasks where only one can be active simultaneously. A *task* is a high-level behavioral sequence as opposed to the low-level actions performed directly by actuators. A reinforcement mechanism is used as a basic learning tool to allow the agents to learn to be more efficient in tasks that are often used. This architecture has been used in the MANTA system to simulate the behavior of ants (Ferber 1996).

Reactive agents do not have representations of their environment and act using a stimulus-response type of behavior; they respond to the present state of the environment in which they are situated.

Reactive agents do not have representations of their environment and act using a stimulus-response type of behavior; they respond to the present state of the environment in which they are situated. They do not take history into account or plan for the future. Through simple interactions with other agents, complex global behavior can emerge. This characteristic is a strength of the approach because the agents do not need to revise their world model as it changes. Thus, robustness and fault tolerance are two of the main properties of reactive systems. A group of agents can complete a task even if one of them fails. However, purely reactive systems suffer from two main limitations: First, because purely reactive agents make decisions based on local information, they cannot take into consideration nonlocal information or predict the effect of their decisions on global behavior. Such myopic behavior could lead to unpredictable and unstable system behavior (Thomas and Sycara 1998; Huberman and Hogg 1988). In reactive systems, the relationship between individual behaviors, environment, and overall behavior is not understandable, which necessarily makes it hard to engineer agents to fulfill specific tasks: One must use a laborious process of experimentation, trial, and error to engineer an agent or an MAS. Despite these disadvantages, reactive systems have advantages of speed (the sophisticated reasoning of deliberative agents can slow them; hence, they are useful in rapidly changing environments).

In fact, for most problems, neither a purely deliberative nor a purely reactive architecture is appropriate, but hybrid architectures can combine aspects of both. Typically, these architectures are realized as a number of software layers, each dealing with a different level of abstraction. Most architectures find three layers sufficient. Thus, at the lowest level in the hierarchy, there is typically a *reactive layer,* which makes decisions about what to do based on raw sensor input. The *middle layer* typically abstracts away from raw sensor input and deals with a knowledge-level view of the agent's environment, typically making use of symbolic representations. The *uppermost level* of the architecture tends to deal with the social aspects of the environment. Coordination with other agents is typically represented in the uppermost layer. The way that the layers interact with one another to produce the global behavior of the agent differs from architecture to architecture (Bonasso et al. 1996; Ferguson 1995, 1992; Müller and Pischel 1994).

The area of agent architectures, particularly layered architectures, continues to be an area of considerable research effort within the multiagent field. For example, there is ongoing work to investigate the appropriateness of various architectures for different environment types. It turns out to be hard to evaluate one agent architecture against another, especially within the context of an MAS.

Organizations

An *organization* provides a framework for agent interactions through the definition of roles, behavior expectations, and authority relations. Organizations are, in general, conceptualized in terms of their structure, that is, the pattern of information and control relations that exist among agents and the distribution of problem-solving capabilities among them. In cooperative problem solving, for example (Corkill and Lesser 1983), a *structure* gives each agent a high-level view of how the group solves problems. The structure should also indicate the connectivity information to the agents so they can distribute subproblems to competent agents.

In open-world environments, agents in the system are not statically predefined but can dynamically enter and exit an organization, which necessitates mechanisms for agent locating. This task is challenging, especially in environments that include large numbers of agents and that have information sources, communication links, and/or agents that might be appearing and disappearing. Researchers have identified different kinds of middle agent (Decker, Sycara, and Williamson 1997) that help agents find others. When an agent is instantiated, it advertises its capabilities to a middle agent. An agent that is looking to find another that possesses a particular capability (for example, can supply particular information or achieve a problem-solving goal) can query a middle agent. In the RETSINA infrastructure, there could be multiple middle agents, not only in type but also in number. For example, protocols have been developed for distributed matchmaking (Jha et al. 1998).

Another perspective in DAI defines organization less in terms of structure and more in terms of current organization theory. For example, Gasser (1986) views an organization as a "particular set of settled and unsettled questions about beliefs and actions through which agents view other agents." In this view, an organization is defined as a set of agents with mutual commitments, global commitments, and mutual beliefs (Bond and Gasser 1988). An organization consists of a group of agents, a set of activities performed by the agents, a set of connections among agents, and a set of goals or evaluation criteria by which the combined activities of the agents are evaluated. The organizational structure imposes constraints on the ways the agents communicate and coordinate. Examples of organizations that have been explored in the MAS literature include the following:

Hierarchy: The authority for decision making and control is concentrated in a single problem solver (or specialized group) at each level in the hierarchy. Interaction is through vertical communication from superior to subordinate agent, and vice versa. Superior agents exercise control over resources and decision making.

Community of experts: This organization is flat, where each problem solver is a specialist in some particular area. The agents interact by rules of order and behavior (Lewis and Sycara 1993; Lander, Lesser, and Connell 1991). Agents coordinate though mutual adjustment of their solutions so that overall coherence can be achieved.

Market: Control is distributed to the agents that compete for tasks or resources through bidding and contractual mechanisms. Agents interact through one variable, price, which is used to value services (Müllen and Wellman 1996; Davis and Smith 1983; Sandholm 1993). Agents coordinate through mutual adjustment of prices.

Scientific community: This is a model of how a pluralistic community could operate (Kornfeld and Hewitt 1981). Solutions to problems are locally constructed, then they are communicated to other problem solvers that can test, challenge, and refine the solution (Lesser 1991).

In open, dynamic environments, the issue of organizational adaptivity is crucial. Organizations that can adapt to changing circumstances by altering the pattern of interactions among the different constituent agents have the potential to achieve coherence in changing and open environments. RETSINA exhibits organizational adaptivity through cooperation mediated by middle agents. RETSINA agents find their collaborators dynamically based on the requirements of the task and on which agents are part of the society at any given time, thus adaptively forming teams on demand (Decker, Williamson, and Sycara 1996a).

Task Allocation

Task allocation is the problem of assigning responsibility and problem-solving resources to an agent. Minimizing task interdependencies has two general benefits regarding coherence: First, it improves problem-solving efficiency by decreasing communication overhead

Organizations are, in general, conceptualized in terms of their structure, that is, the pattern of information and control relations that exist among agents and the distribution of problem-solving capabilities among them.

among the problem-solving agents. Second, it improves the chances for solution consistency by minimizing potential conflicts. In the second case, it also improves efficiency because resolving conflicts can be a time-consuming process.

The issue of task allocation was one of the earliest problems to be worked on in DAI research. On the one extreme, the designer can make all the task assignments in advance, thus creating a nonadaptive problem-solving organization. This approach is limiting and inflexible for environments with a high degree of dynamism, openness, and uncertainty. However, one can do task allocation dynamically and flexibly. The issue of flexible allocation of tasks to multiple agents received attention early on (Davis and Smith 1983). Davis and Smith's work resulted in the well-known CONTRACT NET PROTOCOL (CNP). In this protocol, agents can dynamically take two roles: (1) manager or (2) contractor. Given a task to perform, an agent first determines whether it can break it into subtasks that can be performed concurrently. It utilizes the protocol to announce the tasks that could be transferred and requests bids from agents that could perform any of these tasks. An agent that receives a task-announcement message replies with a bid for the task, indicating how well it thinks it can perform the task. The contractor collects the bids and awards the task to the best bidder. The CNP allows nodes to broadcast bid requests to all others. The CNP was subsequently used to control factory floor operations (Parunak 1987). Although CNP was considered by Smith and Davis and many DAI researchers to be a negotiation principle, it is a coordination method for task allocation. CNP enables dynamic task allocation, allows agents to bid for multiple tasks at a time, and provides natural load balancing (busy agents need not bid). Its limitations are that it does not detect or resolve conflicts, the manager does not inform nodes whose bids have been refused, agents cannot refuse bids, there is no preemption in task execution (time-critical tasks might not be attended to), and it is communication intensive. Extensions to CNP have been made by Sandholm and Lesser (1995), where decommitment penalties were introduced, and by Sycara (1997), where the theory of financial option pricing has been used to achieve flexible contracting schemes in uncertain environments.

Multiagent Planning

Agents can improve coherence by planning their actions. Planning for a single agent is a process of constructing a sequence of actions considering only goals, capabilities, and environmental constraints. However, planning in an MAS environment also considers the constraints that the other agents' activities place on an agent's choice of actions, the constraints that an agent's commitments to others place on its own choice of actions, and the unpredictable evolution of the world caused by other unmodeled agents.

Most early work in DAI has dealt with groups of agents pursuing common goals (for example, Lesser [1991]; Lesser, Durfee, and Corkill [1989]; Durfee [1988]; Conry, Meyer and Lesser [1988]; Cammarata, McArthur, and Steeb [1983]). Agent interactions are guided by cooperation strategies meant to improve their collective performance. Most work on multiagent planning assumes an individual sophisticated agent architecture that enables them to do rather complex reasoning. Early work on distributed planning took the approach of complete planning before action. To produce a coherent plan, the agents must be able to recognize subgoal interactions and avoid them or resolve them. Early work by Georgeff (Rao and Georgeff 1991) included a static agent to recognize and resolve such interactions. The agents sent this agent their plan; the agent examined them for critical regions where, for example, contention for resources could cause them to fail. The agent then inserted synchronization messages (akin to operating system semaphores) so that one agent would wait till the resource was released by another agent. In the work by Cammarata on air-traffic control (Steeb et al. 1988; Cammarata, McArthur, and Steeb 1983), the synchronizing agent was dynamically assigned according to different criteria, and it could alter its plan to remove the interaction (avoid collision). Another approach to resolving subproblem interdependencies that result from an incomplete set of data for each agent that could allow it to solve a subproblem completely is the FUNCTIONALLY ACCURATE MODEL (FA/C) (Lesser 1991). In the FA/C model, agents do not need to have all the necessary information locally to solve their subproblems but instead interact through the asynchronous, coroutine exchange of partial results. With the FA/C model, a series of sophisticated distributed control schemes for agent coordination were developed, such as the use of static metalevel information specified by an organizational structure and the use of dynamic metalevel information developed in partial global planning (PGP) (Durfee 1987).

PGP is a flexible approach to coordination that does not assume any particular distribution of subproblems, expertise, or other resources but, instead, lets nodes coordinate in response to current situations (Durfee 1987). Agent interactions take the form of communicating plans and goals at an appropriate level of abstraction. These communications enable a receiving agent to form expectations about the future behavior of a sending agent, thus improving agent predictability and network coherence (Durfee 1988). Because agents are cooperative, the recipient agent uses the information in the plan to adjust its own local planning appropriately, so that the common planning goals (and planning effectiveness criteria) are met. Besides their common PGPs, agents also have some common knowledge about how and when to use PGPs. Decker and Lesser (1995) addressed some of the limitations of the PGP by creating a generic PGP-based framework called TAEMS to handle issues of real time (for example, scheduling to deadlines) and metacontrol (for example, to obviate the need to do detailed planning at all possible node interactions). TAEMS has been used as a framework for evaluation of coordination algorithms.

Another direction of research in cooperative multiagent planning has been focused on modeling teamwork explicitly. Explicit modeling of teamwork is particularly helpful in dynamic environments where team members might fail or be presented with new opportunities. In such situations, it is necessary that teams monitor their performance and reorganize based on the situation.

The *joint-intentions framework* (Cohen and Levesque 1990) focuses on characterizing a team's mental state, called a *joint intention*. A team jointly intends a team action if the team members are jointly committed to completing the team action while mutually believing they were doing it. A *joint commitment* is defined as a joint persistent goal. To enter into a joint commitment, all team members must establish appropriate mutual beliefs and commitments, which is done through an exchange of request and confirm speech acts (Cohen and Levesque 1990). The commitment protocol synchronizes the team in that all members simultaneously enter into a joint commitment toward a team task. In addition, all team members must consent, using confirmation, to the establishment of a joint commitment goal. If a team member refuses, negotiation could be used; however, how it is done remains an open issue.

The model of SHAREDPLAN (Grosz and Kraus 1996; Grosz and Sidner 1990) is not based on a joint mental attitude but rather on a new mental attitude intending that an action be done. *Intending* is defined using a set of axioms that guide a teammate to take action or enter into communication that enables or facilitates its teammates to perform assigned tasks. COLLAGEN (Rick and Sidner 1997) is a prototype toolkit, which has its origins in SHAREDPLAN, and is applied to building a collaborative interface agent that helps with air-travel arrangements. Jennings (1995) presented a framework called joint responsibility based on a joint commitment to a team's joint goal and a joint recipe commitment to a common recipe. This model was implemented in the GRATE system.

Tambe (1997) presents a model of teamwork, called STEAM (a shell for TEAMWORK), based on enhancements to the SOAR architecture (Newell 1990), plus a set of about 300 domain-independent SOAR rules. Based on the teamwork operationalized in STEAM, three applications have been implemented, two that operate in a commercially available simulation for military training and the third that is part of ROBOCUP synthetic soccer. STEAM uses a hybrid approach that combines joint intentions (Cohen and Levesque 1990) but also uses partial SHAREDPLANS (Grosz and Kraus 1996).

Increasingly, the emphasis of multiagent planning has been on flexible communication and action execution in complex, dynamic environments, including agents that might be hostile or at least self-interested (Veloso et al. 1997) and perform well in dynamically changing environments (Kinny et al. 1992).

Recognizing and Resolving Conflicts

Because MASs lack global viewpoints, global knowledge, and global control, there is the potential for disparities and inconsistencies in agents' goals, plans, knowledge, beliefs, and results. To achieve coherent problem solving, these disparities must be recognized and resolved. Disparities can be resolved by making an agent omniscient so it can see the states of all agents and determine where the disparities lie and how to resolve them. This approach is limiting because it makes this agent a bottleneck and a single point of failure. To detect and correct disparities and conflicts using only local perspective is difficult. To facilitate detection and resolution of conflicts, agents can rely on models of the world and other agents. Disparity resolution can be influenced by the organizational structure of the agent society and an agent's role within it, the kinds of models an agent has, and the agent's reasoning algorithms.

The main approach for resolving disparities in an MAS is negotiation. Before I treat negotiation in some detail, I present some less often used approaches that are organized in Gasser (1992). They include (1) assumption surfacing, where inconsistent propositions can be backed up to their assumptions (Huhns and Bridgeland 1991; Mason and Johnson 1989); (2) evidential reasoning and argumentation, where it might be possible to construct arguments in support of a particular perspective (Loui 1987; Hewitt 1986, Kornfeld and Hewitt 1981; Lesser and Corkill 1981), or it might be possible to construct persuasive arguments to change the intentions, beliefs, preferences, and actions of a persuadee (Sycara 1990b) so that effective plans and solutions can be produced; (3) constraint relaxation, where conflicting constraints can be resolved by relaxing them (Liu and Sycara 1997; Sycara et al. 1991) or reformulating the problem to eliminate the constraints (Sycara 1991); and (4) social norms that impose some sort of common standard of behavior, which when adhered to can lead to conflict avoidance (Castelfranchi, Miceli and Cesta 1992). Social norms and standards, although helping avoid conflicts, can impede adaptation.

Negotiation is seen as a method for coordination and conflict resolution (for example, resolving goal disparities in planning, resolving constraints in resource allocation, resolving task inconsistencies in determining organizational structure). Negotiation has also been used as a metaphor for communication of plan changes, task allocation (for example, CNP), or centralized resolution of constraint violations. Hence, negotiation is almost as ill defined as the notion of agent. I give here what I consider as the main characteristics of negotiation that are necessary for developing applications in the real world: (1) the presence of some sort of conflict that must be resolved in a decentralized manner by (2) self-interested agents under conditions of (3) bounded rationality and (4) incomplete information. Furthermore, the agents communicate and iteratively exchange proposals and counterproposals.

My PERSUADER system (1990b, 1988, 1987) and work by Rosenschein (Rosenschein and Zlotkin 1994; Rosenschein and Genesereth 1985; Zlotkin and Rosenschein 1991) are the first AI research on negotiation among self-interested agents. The two approaches differ in their assumptions, motivations, and operationalism. The work of Rosenschein was based on game theory. Utility is the single issue that the agents consider, and the agents are assumed omniscient. Utility values for alternative outcomes are represented in a payoff matrix that is common knowledge to both parties in the negotiation. Each party reasons about and chooses the alternative that will maximize its utility. Despite the mathematical elegance of game theory, game-theoretic models suffer from restrictive assumptions that limit their applicability to realistic problems.[1] Real-world negotiations are conducted under uncertainty, they involve multiple criteria rather than a single utility dimension, the utilities of the agents are not common knowledge but private, and the agents are not omniscient.

PERSUADER (Sycara 1990a) is an implemented system that involves three agents—(1) a union, (2) a company, and (3) a mediator—and it operates in the domain of labor negotiation. It is inspired by human negotiation. It models iterative exchange of proposals and counterproposals for the parties to reach agreement. The negotiation involves multiple issues, such as wages, pensions, seniority, and subcontracting. Each agent's multidimensional utility model is private (rather than common) knowledge. Belief revision to change the agents' utilities so that agreement can be reached is done through persuasive argumentation (Sycara 1990b). In addition, case-based learning is also incorporated in the model. The negotiation model of PERSUADER has also been applied to the domain of concurrent engineering (Lewis and Sycara 1993). Parsons and Jennings (1996) have followed the formalism described in Kraus, Nirke, and Sycara (1993) to construct arguments to evaluate proposals and counterproposals in negotiation.

Work by Kraus, Wilkenfeld, and Zlotkin (1995) focuses on the role of time in negotiation. Using a distributed mechanism, the agents conduct negotiation and can reach efficient agreements without delays. It is also shown that the individual approach of each agent toward the negotiation time affects and even determines the final agreement that is reached. The main conclusion is that delaying agreements causes inefficiency in the negotiation.

Liu and Sycara (1994) and Garrido and Sycara (1996) have modeled negotiation as a constraint-relaxation process where the agents are self-interested in the sense that they would like to achieve an agreement that gives them the highest utility but are also cooperative in the sense that they are willing to accept lower utility to facilitate reaching an agreement. The agents communicate their constraints through proposals and counterproposals. The application domain of the work has been a distributed meeting scheduling. In Garrido and Sycara (1996), experimental results were obtained to study the scheduling efficiency and quality of the agreed-on final schedule under various conditions of information privacy and preference structures of the participants.

Because electronic commerce is rapidly becoming a reality, the need for negotiation techniques that take into consideration the complexities of the real world, such as incomplete information, multiple negotiation issues, negotiation deadlines, and the ability to break contracts, will critically be needed. Work in nonbinding contracts includes Sandholm and Lesser (1995), where decommitment penalties were introduced into the CNP, and my work (Sycara 1997), where financial option pricing has been used to model the value of a contingent contract and calculate optimal values of different contracting parameters of a contingent contract, such as when to give a contract to a contractee, when to break a contract, and which contract to accept of a set of offered contracts.

Modeling Other Agents

Agents can increase the accuracy and efficiency of their problem solving if they are given knowledge about other agents or the ability to model and reason about others. They can utilize this knowledge to predict possible conflicts and interpret, explain, and predict the other agents' actions under varying circumstances. In addition, modeling others provides information to an agent about others' knowledge, beliefs, and goals that are useful in planning when and what to communicate or allowing coordination without communication. The ability to model others increases an agent's flexibility. Instead of operating on the basis of a fixed protocol of interaction, modeling others allows the agent to change the pattern of interaction. Having a model allows an agent to infer events or behaviors of the other agent that it cannot sense directly. Several typical components of agent models are commitments, capabilities, resources, beliefs, plans, and organizational knowledge. We saw already how explicit commitments to joint activity are the cornerstone of models of teamwork. In my own work on the RETSINA multiagent infrastructure, commitment of an agent to performing a task is not communicated explicitly to others but is implicit in its advertisement to a middle agent. An agent's advertisement describes its capabilities, that is, the set of services it can provide to others. For example, the advertisement of an information agent expresses the set of queries that the agent is capable of answering. In this way, an agent that needs a particular service can, through asking a middle agent, find out who is capable of providing the service and can contact the service provider(s). Thus, an agent's advertisement is the model of itself it makes available to the agent society. Thus, for example, instead of agent *X* either having a model of agent *Y* built into it at design time or having to infer it, agent *Y* makes its capability model available to agent *X* through an advertisement to a middle agent. These two ways would be infeasible and extremely time consuming for agents in large-scale and open environments such as the internet. Providing a model of oneself, however, is a solution that is scalable and works in an open environment. If the agent goes down, it "unadvertises"; that is, it lets it be known that it is no longer a member of the society.

Managing Communication

Agents can improve the coherence of their problem solving by planning the content, amount, type, and timing of the communication they exchange. It has been noted (Durfee, Lesser, and Corkill 1987) that using abstraction and metalevel information (for example, organizational knowledge) is helpful because they help decrease communication overhead. In dynamic and open environments, inhabited by heterogeneous agents, additional issues need to be faced. The most prominent among them is agent interoperability.

Agents are populating the internet at a rapid pace. These agents have different functions (capabilities). There can, however, be many agents with the same functions (for example, many information agents provide financial news). Agents can increase their problem-solving scope by cooperation. In an open environment, heterogeneous agents that would like to coordinate with each other (either cooperate or negotiate, for example) face two major challenges: First, they must be able to find each other (in an open environment, agents might appear and disappear unpredictably), and second, they must be able to interoperate.

To address the issue of finding agents in an open environment such as the internet, middle agents (Decker, Williamson, and Sycara 1996b) have been proposed. Different agent types were identified and implemented (Decker, Sycara, and Williamson 1997). These types include matchmakers or yellow page agents that match advertisements to requests for advertised capabilities, blackboard agents that collect requests, and brokers that process both. In preliminary experiments (Decker, Sycara, and Williamson 1997), it was seen that the behaviors of each type of middle agent have certain performance characteristics; for example, although brokered systems are more vulnerable to certain failures, they are also able to cope more quickly with a rapidly fluctuating agent work force. Middle agents are advantageous because they allow a system to operate robustly in the face of agent appearance and disappearance and intermittent communications.

To allow agents to interoperate, communication languages, such as KQML (Finin et al. 1994) and the one developed by Cohen and his colleagues (Smith and Cohen 1996) that provide a set of performatives based on speech acts, have been designed. Although such performatives can characterize message types, efficient languages to express message content that allows agents to understand each other have not been demonstrated effectively (although KIF [Genesereth and Ketchpel 1994] has been proposed). The *ontology problem,* that is, how agents can share meaning, is still open (Gruber 1993).

Economics-based approaches, such as market mechanisms, are becoming increasingly attractive to MAS researchers because of the ready availability of underlying formal models and their potential applicability in internet-based commerce.

Managing Resources

Another critical issue is effective allocation of limited resources to multiple agents. For example, we have all experienced large time lags in response to internet queries because of network congestion.

Various approaches have been developed for effective resource allocation to multiple agents. Some of them hail from operations research–based techniques for single-agent scheduling, and others use market-oriented approaches. In the first category of approaches, I mention *distributed constraint heuristic search* (Sycara et al. 1991), which combines decentralized search with constraint satisfaction and optimization. The method relies on (1) a set of variable and value-ordering heuristics that quantify several characteristics of the space being searched and (2) a communication protocol that allows the agents to coordinate in an effective manner. Another distributed scheduling model exploits a large number of simple agents by partitioning problem constraints and assigning them to specialized agent classes. This methodology was applied to solve job-shop–scheduling constraint-satisfaction and constraint-optimization problems (Liu and Sycara 1997, 1995a, 1995b) with good results on standard benchmark problems from the operations research literature in problem scalability (as many as 5000 agents), computational efficiency, and high-solution quality under different optimization criteria (for example, the minimization of makespan, weighted tardiness, and so on) (Liu and Sycara 1993).

Economics-based mechanisms have been utilized in MASs to address problems of resource allocation (the central theme of economic research). Economics-based approaches, such as market mechanisms, are becoming increasingly attractive to MAS researchers because of the ready availability of underlying formal models and their potential applicability in internet-based commerce. In economics-based approaches, agents are assumed to be self-interested utility maximizers. In markets, agents that control scarce resources (labor, raw materials, goods, money) agree to share by exchanging some of their respective resources to achieve some common goal. Resources are exchanged with or without explicit prices. Market mechanisms have been used for resource allocation (Müllen and Wellman 1996; Huberman and Clearwater 1995; Sandholm 1993). Markets assume that the exchange prices are publicly known. In auctions, there is a central auctioneer through which coordination happens. Hence, the agents exchange minimal amounts of information.

It is probable that in the future, most agents will be self-interested. A *self-interested agent* simply chooses a course of action that maximizes its own utility. In a society of self-interested agents, it is desired that if each agent maximizes its local utility, the whole society exhibits good behavior; in other words, good local behavior implies good global behavior. The goal is to design mechanisms for self-interested agents such that if agents follow these mechanisms, the overall system behavior will be acceptable, which is called *mechanism design*. Many problems face such a society of self-interested agents: First, agents might overuse and, hence, congest a shared resource, such as a communications network. This problem is called the *tragedy of the commons* (Hardin 1968). Generally, the problem of tragedy of commons is solved by pricing or taxing schemes.

Second, a society of self-interested computational agents can exhibit oscillatory or chaotic behavior (Thomas and Sycara 1998; Huberman and Hogg 1988). The experimental results show that imperfect knowledge suppresses oscillatory behavior at the expense of reducing performance. Moreover, systems can remain in nonoptimal metastable states for extremely long times before getting to the globally optimal state. In Thomas and Sycara (1998), a similar problem is considered. Two approaches are evaluated: (1) heterogeneous preferences and (2) heterogeneous transaction costs; empirically, the transaction cost case provides stability with near-optimal payoffs under certain conditions.

Third, agents might be untruthful or deceitful to increase their individual utility. Lying and deceitfulness can have harmful effects on the whole society. Mechanism design techniques have been reported that make it beneficial for agents to report the truth (Myerson 1989). Another body of research explores the effective allocation of computational resources and load-balancing issues. In Shehory, Jha, and Sycara (1997), agent-cloning and agent-merging techniques were developed as a way to mitigate agent overloading and promote system load balancing.

Adaptation and Learning

MAS adaptivity to changing circumstances by altering the problem-solving behavior of individual agents or the patterns of agent interactions provides the potential for increasing problem-solving coherence.

Learning in a multiagent environment is complicated by the fact that the environment effectively changes as other agents learn. Moreover, other agents' actions are often not directly observable, and the action taken by the learning agent can strongly bias which range of behaviors is encountered. In Hu and Wellman (1996), an agent's belief process is characterized in terms of conjectures about the effect of its actions. A conjectural equilibrium is then defined where all agents' expectations are realized, and each agent responds optimally to its expectations. An MAS is presented where an agent builds a model of the response of others. Reported experimental results show that depending on the starting point, an agent might be better or worse off than had it not attempted to learn a model of the other agents. Such equivocal results have also been observed in computational ecosystems (Huberman and Hogg 1988), where a large number of simple agents operate asynchronously without central control and compete for resources, with incomplete knowledge and delayed information. Kephart, Hogg, and Huberman (1989) performed experimental simulations and verified some of the previous theoretical results; thus, for certain parameter settings, these systems exhibit behavior characterized by fixed points, oscillation, and even chaos. Complex behavior thus can be exhibited by simple computational ecosystems. Enhancing the decision-making abilities of some of the individuals in the system can either improve or severely degrade overall system performance.

Recently, there has been increasing interest in integrating learning into the negotiation process (for example, Sen [1996]). Zeng and Sycara (1997) have developed an economic bargaining negotiation model, BAZAAR, where the agents are self-interested. The model emphasizes the learning aspects. The agents keep histories of their interactions and update their beliefs, using Bayesian updating, after observing their environment and the behavior of the other negotiating agents. The benefits of learning, if any, on the individual utilities of agents, as well as the overall (joint) system utility, were examined. The experimental results suggest that (1) when all agents learn, the joint system utility is near optimal, and agents' individual utilities are similar; (2) when no agent learns, the agents' individual utilities are almost equal, but the joint utility is low (much lower than in the all-agents- learn condition); and (3) when only one agent learns, its individual utility increases at the expense of both the individual utility of the other agents as well as the overall joint utility of the system (that is, only one agent learning has a harmful overall effect) (Zeng and Sycara 1998, 1997). A survey of multiagent learning can be found in Stone and Veloso (1997).

Multiagent System Applications

The first MAS applications appeared in the mid-1980s and increasingly cover a variety of domains, ranging from manufacturing to process control, air-traffic control, and information management. I give here a few representative examples. For a more detailed description of agent-based applications, see Jennings et al. (1998) and Chaib-draa (1995).

One of the earliest MAS applications was *distributed vehicle monitoring* (DVMT) (Durfee 1996, 1987; Durfee and Lesser 1989), where a set of geographically distributed agents monitor vehicles that pass through their respective areas, attempt to come up with interpretations of what vehicles are passing through the global area, and track vehicle movements. The DVMT has been used as an MAS test bed.

Parunak (1987) describes the YAMS (YET ANOTHER MANUFACTURING SYSTEM) system, which applies the CNP to manufacturing control. The basic problem can be described as follows: A manufacturing enterprise is modeled as a hierarchy of functionally specific work cells. These work cells are further grouped into flexible manufacturing systems (FMSs) that collectively constitute a factory. The goal of YAMS is to efficiently manage the production process of these factories. To achieve this complex task, YAMS adopts a multiagent approach, where each factory and factory component is represented as an agent. Each agent has a collection of plans representing its capabilities. The CNP allows tasks (that is, production orders) to be delegated to individual factories, and from individual factories down to FMSs, and then to individual work cells. Other systems in this area include those for configuration design of manufacturing products (Darr and Birmingham 1996) and collaborative design (Cutkosky et al. 1993).

The best-known MAS for process control is ARCHON, a software platform for building MASs and an associated methodology for building applications with this platform (Jennings, Corera, and Laresgoiti 1995). ARCHON has been applied in several process-control applications, including electricity transportation management (the application is in use in northern Spain) and particle accelerator control. ARCHON also has the distinction of being one of the world's earliest field-tested MASS. Other agent-based process-control systems have been written for monitoring and diagnosing faults in nuclear power plants (Wang and Wang 1997), spacecraft control (Schwuttke and Quan 1993), and climate control (Huberman and Clearwater 1995).

Ljunberg and Lucas (1992) describe a sophisticated agent-based air-traffic control system known as OASIS. In this system, which is undergoing field trials at the Sydney, Australia, airport, agents are used to represent both aircraft and the various air-traffic control systems in operation. The agent metaphor thus provides a useful and natural way of modeling real-world autonomous components. As an aircraft enters Sydney airspace, an agent is allocated for it, and the agent is instantiated with the information and goals corresponding to the real-world aircraft. For example, an aircraft might have a goal to land on a certain runway at a certain time. Air-traffic control agents are responsible for managing the system. OASIS is implemented using the Australian Artificial Intelligence Institute's own BDI model of agency (DMARS).

The WARREN financial portfolio management system (Sycara et al. 1996) is an MAS that integrates information finding and filtering from the internet in the context of supporting a user manage his/her financial portfolio. The system consists of agents that cooperatively self-organize to monitor and track stock quotes, financial news, financial analyst reports, and company earnings reports to appraise the portfolio owner of the evolving financial picture. The agents not only answer relevant queries but also continuously monitor internet information sources for the occurrence of interesting events (for example, a particular stock has gone up past a threshold) and alert the portfolio manager agent or the user. WARREN also includes agents that analyze user buy and sell decisions with respect to asset allocations and risk (Sycara, Decker, and Zeng 1998).

In addition, there are a variety of MAS applications in telecommunications. In one such application (Weihmayer and Velthuijsen 1994), Griffeth and Velthuijsen use negotiating agents to tackle the feature interaction problem by utilizing negotiating agents to represent the different entities that are interested in the set up of a call. When conflicts are detected, the agents negotiate with one another to resolve them so that an acceptable call configuration can be established. Other problems for which agent-based systems have been constructed include network control, transmission and switching, service management, and network management.

Conclusions

Designing and building agent systems is difficult. They have all the problems associated with building traditional distributed, concurrent systems and have the additional difficulties that arise from having flexible and sophisticated interactions between autonomous problem-solving components. The big question then becomes one of how effective MASs can be designed and implemented.

At this time, there are two major technical impediments to the widespread adoption of multiagent technology: (1) the lack of a systematic methodology enabling designers to clearly specify and structure their applications as MASs and (2) the lack of widely available industrial-strength MAS toolkits.[2] Flexible sets of tools are needed that enable designers to specify an agent's problem-solving behavior, specify how and when agents should interact, and visualize and debug the problem-solving behavior of the agents and the entire system.

The other major impediment to the widespread adoption of agent technology has a social, as well as a technical, aspect. For individuals to be comfortable with the idea of delegating tasks to agents, they must first trust them (Bradshaw 1997; Maes 1994). The process of mutual adjustment between user and agents (both in terms of the agent learning user preferences but also in terms of the user learning agents' capabilities and limitations) takes time. During this period, agents must strike a balance between continually seeking guidance (and needlessly distracting the user) and never seeking guidance (and exceeding their authority).

Acknowledgments

This work on agent-based systems has been supported by Office of Naval Research grant N-00014-96-1-1222, Advanced Research Projects Agency contract F33615-93-1-1330, and National Science Foundation grant IRI-9612131.

Notes

[1] It should be noted that some of the very recent game-theoretic models are directly motivated by considerations of dropping or relaxing some of these assumptions. Although there has been interesting progress reported in the literature (for example, Jordan [1992]), the fundamental framework and methodology of game theory remains almost the same, and it might be too early to tell whether these new results will reshape the current game-theoretic framework.

[2] Existing technologies such as CORBA are at a low level and, thus, unable to provide the support needed for the structuring of flexible multiagent systems.

References

Bonasso, R. P.; Kortenkamp, D.; Miller, D. P.; and Slack, M. 1996. Experiences with an Architecture for Intelligent, Reactive Agents. In *Intelligent Agents II*, eds. M. Wooldridge, J. P. Muller, and M. Tambe, 187–202. Lecture Notes in Artificial Intelligence 1037. New York: Springer-Verlag.

Bond, A. H., and Gasser, L. 1988. *Readings in Distributed Artificial Intelligence.* San Francisco, Calif.: Morgan Kaufmann.

Bradshaw, J. 1997. *Software Agents.* Menlo Park, Calif.: AAAI Press.

Brooks, R. A. 1991. Intelligence without Representation. *Artificial Intelligence* 47(1–3): 139–159.

Cammarata, S.; McArthur, D.; and Steeb, R. 1983. Strategies of Cooperation in Distributed Problem Solving. In Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83), 767–770. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Castelfranchi, C.; Miceli, M.; and Cesta, A. 1992. Dependence Relations among Autonomous Agents. In *Decentralized Artificial Intelligence,* eds. E. Werner and Y. Demazeau, 215–231. Amsterdam, The Netherlands: Elsevier/North-Holland.

Chaib-draa, B. 1995. Industrial Applications of Distributed AI. *Communications of the ACM* 38(11): 49–53.

Chaib-draa, B.; Moulin, B.; Mandiau, R.; and Millot, P. 1992. Trends in Distributed Artificial Intelligence. *Artificial Intelligence Review* 6(1): 35–66.

Cohen, P. R., and Levesque, H. J. 1990. Intention Is Choice with Commitment. *Artificial Intelligence* 42(2–3): 213–261.

Conry, S. E.; Meyer, R. A.; and Lesser, V. R. 1988. Multistage Negotiation in Distributed Planning. In *Readings in Distributed Artificial Intelligence,* eds. A. H. Bond and L. Gasser, 367–384. San Francisco, Calif.: Morgan Kaufmann.

Corkill, D. D., and Lesser, V. R. 1983. The Use of Metalevel Control for Coordination in a Distributed Problem- Solving Network. In Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83), 767–770. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Cutkosky, M.; Fikes, R.; Engelmore, R.; Genesereth, M.; Mark, W.; Gruber, T.; Tenenbaum, J.; and Weber, J. 1993. PACT: An Experiment in Integrating Concurrent Engineering Systems. *IEEE Transactions on Computers* 26(1): 28–37.

Darr, T., and Birmingham, W. 1996. An Attribute-Space Representation and Algorithm for Concurrent Engineering. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing* 10(1): 21–35.

Davis, R., and Smith, R. G. 1983. Negotiation as a Metaphor for Distributed Problem Solving. *Artificial Intelligence* 20(1): 63–100.

Decker, K., and Lesser, V. 1995. Designing a Family of Coordination Algorithms. In *Proceedings of the First International Conference on Multiagent Systems,* 73–80. Menlo Park, Calif.: AAAI Press.

Decker, K.; Pannu, A.; Sycara, K.; and Williamson, M. 1997. Designing Behaviors for Information Agents. In Proceedings of the First International Conference on Autonomous Agents (Agents-97), 404–412. New York: Association of Computing Machinery.

Decker, K.; Sycara, K.; and Williamson, M. 1997. Middle Agents for the Internet. In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97), 578–583. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Decker, K.; Williamson, M.; and Sycara, K. 1996a. Intelligent Adaptive Information Agents. Paper presented at the AAAI-96 Workshop on Intelligent Adaptive Agents, 4–7 August, Portland, Oregon.

Decker, K.; Williamson, M.; and Sycara, K. 1996b. Matchmaking and Brokering. In *Proceedings of the Second International Conference on Multiagent Systems,* 432. Menlo Park, Calif.: AAAI Press.

Dent, L.; Boticario, J.; McDermott, J.; Mitchell, T.; and Zabowski, D. 1992. A Personal Learning Apprentice. In Proceedings of the Tenth National Conference on Artificial Intelligence, 96–103. Menlo Park, Calif.: American Association for Artificial Intelligence.

Drogul, A., and Ferber, J. 1992. From Tom Thumb to the Dockers: Some Experiments with Foraging Robots. In *From Animals to Animats: Second Conference on Simulation of Adaptive Behavior,* eds. H. R. J. Meyer and S. Wilson. Cambridge, Mass.: MIT Press.

Durfee, E. 1996. Planning in Distributed Artificial Intelligence. In *Foundations of Distributed Artificial Intelligence,* eds. G. M. P. O'Hare and N. R. Jennings, 231–246. New York: Wiley.

Durfee, E. H. 1988. *Coordination of Distributed Problem Solvers.* Boston: Kluwer Academic.

Durfee, E. H. 1987. A Unified Approach to Dynamic Coordination: Planning Actions and Interactions in a Distributed Problem Solving Network, Ph.D. dissertation, Department of Computer and Information Science, University of Massachusetts.

Durfee, E. H., and Lesser, V. 1989. Negotiating Task Decomposition and Allocation Using Partial Global Planning. In *Distributed Artificial Intelligence, Volume 2,* eds. L. Gasser and M. Huhns, 229–244. San Francisco, Calif.: Morgan Kaufmann.

Durfee, E. H.; Lesser, V. R.; and Corkill, D. D. 1987. Coherent Cooperation among Communicating Problem Solvers. *IEEE Transactions on Computers* C-36(11): 1275–1291.

Ferber, J. 1996. Reactive Distributed Artificial Intelligence: Principles and Applications. In *Foundations of Distributed Artificial Intelligence,* eds. G. O-Hare and N. Jennings, 287–314. New York: Wiley.

Ferguson, I. A. 1995. Integrated Control and Coordinated Behavior: A Case for Agent Models. In *Intelligent Agents: Theories, Architectures, and Languages,* eds. M. Wooldridge and N. R. Jennings, 203–218. Lecture Notes in Artificial Intelligence, Volume 890. New York: Springer-Verlag.

Ferguson, I. A. 1992. TOURINGMACHINES: An Architecture for Dynamic, Rational, Mobile Agents, Ph.D. dissertation, Technical report 273, Computer Laboratory, University of Cambridge.

Finin, T.; Fritzson, R.; McKay, D.; and McEntire, R. 1994. KQML as an Agent Communication Language. In Proceedings of the Third International Conference on Information and Knowledge Management (CIKM94), 456–463. New York: Association of Computing Machinery.

Garrido, L., and Sycara, K. 1996. Multiagent Meeting Scheduling: Preliminary Experimental Results. In *Proceedings of the Second International Conference on Multiagent Systems,* 95–102. Menlo Park, Calif.: AAAI Press.

Gasser, L. 1992. An Overview of DAI. In *Distributed Artificial Intelligence: Theory and Praxis,* eds. N. Avouris and L. Gasser, 9–30. Boston: Kluwer Academic.

Gasser, L. 1991. Social Conceptions of Knowledge and Action. *Artificial Intelligence* 47(1–3): 107–138.

Gasser, L. 1986. The Integration of Computing and Routine Work. *ACM Transactions on Office Information Systems* 4(3): 205–225.

Genesereth, M. R., and Ketchpel, S. P. 1994. Software Agents. *Communications of the ACM* 37(7): 48–53.

Grosz, B., and Kraus, S. 1996. Collaborative Plans for Complex Group Actions. *Artificial Intelligence* 86(2): 269–357.

Grosz, B., and Sidner, C. 1990. Plans for Discourse. In *Intentions in Communication,* eds. P. Cohen, J. Morgan, and M. Pollack, 417–444. Cambridge Mass.: MIT Press.

Gruber, T. R. 1993. A Translation Approach to Portable Ontologies. *Knowledge Acquisition* 5(2): 199–220.

Hardin, G. 1968. The Tragedy of Commons. *Science* 162(6): 1243–1248.

Hewitt, C. 1986. Offices Are Open Systems. *ACM Transactions of Office Automation Systems* 4(3): 271–287.

Hu, J., and Wellman, M. P. 1996. Self-Fulfilling Bias in Multiagent Learning. In *Proceed-*

ings of the Second International Conference on Multiagent Systems, 118–125. Menlo Park, Calif.: AAAI Press.

Huberman, B., and Clearwater, S. H. 1995. A Multiagent System for Controlling Building Environments. In *Proceedings of the First International Conference on Multiagent Systems,* 171–176. Menlo Park, Calif.: AAAI Press.

Huberman, B. A., and Hogg, T. 1988. The Behavior of Computational Ecologies. In *The Ecology of Computation,* ed. B. A. Huberman. Amsterdam, The Netherlands: North-Holland.

Huhns, M., and Bridgeland, D. 1991. Multiagent Truth Maintenance. *IEEE Transactions on Systems, Man, and Cybernetics* 216(6): 1437–1445.

Huhns, M., and Singh, M. 1997. *Readings in Agents.* San Francisco, Calif.: Morgan Kaufmann.

Jennings, N. 1995. Controlling Cooperative Problem Solving in Industrial Multiagent Systems Using Joint Intention. *Artificial Intelligence* 75(2): 195–240.

Jennings, N. R.; Corera, J. M.; and Laresgoiti, I. 1995. Developing Industrial Multiagent Systems. In *Proceedings of the First International Conference on Multiagent Systems,* 423–430. Menlo Park, Calif.: AAAI Press.

Jennings, N.; Sycara, K.; and Wooldridge, M. 1998. A Roadmap for Agent Research and Development. *Autonomous Agents and Multiagent Systems* 1(1). Forthcoming.

Jha, S.; Chalasani, P.; Shehory, O.; and Sycara, K. 1998. A Formal Treatment of Distributed Matchmaking. In Proceedings of the Second International Conference on Autonomous Agents (Agents '98). New York: Association of Computing Machinery. Forthcoming.

Jordan, J. S. 1992. The Exponential Covergence of Bayesian Learning in Normal Form Games. *Games and Economic Behavior* 4(2): 202–217.

Kephart, J. O.; Hogg, T.; and Huberman, B. A. 1989. Dynamics of Computational Ecosystems: Implications for DAI. In *Distributed Artificial Intelligence, Volume 2,* eds. L. Gasser and M. Huhns, 79–96. San Francisco, Calif.: Morgan Kaufmann.

Kinny, D.; Ljungberg, M.; Rao, A.; Sonenberg, E.; Tidhard, G.; and Werner, E. 1992. Planned Team Activity. In *Artificial Social Systems,* eds. C. Castelfranchi and E. Werner. New York: Springer-Verlag.

Kornfeld, W. A., and Hewitt, C. E. 1981. The Scientific Community Metaphor. *IEEE Transactions on Systems, Man, and Cybernetics* 11(1): 24–33.

Kraus, S.; Nirke, M.; and Sycara, K. 1993. Reaching Agreements through Argumentation: A Logical Model. Paper presented at the Twelfth International Workshop on Distributed Artificial Intelligence (IWDAI-93), 19–21 May, Hidden Valley, Pennsylvania.

Kraus, S.; Wilkenfeld, J.; and Zlotkin, G. 1995. Multiagent Negotiation under Time Constraints. *Artificial Intelligence* 75(2): 297–345.

Lander, S.; Lesser, V. R.; and Connell, M. E. 1991. Conflict-Resolution Strategies for Cooperating Expert Agents. In *CKBS-90, Proceedings of the International Working Conference on Cooperating Knowledge-Based Systems,* ed. S. M. Deen, 183–200. New York: Springer-Verlag.

Lesser, V. R. 1991. A Retrospective View of FA/C Distributed Problem Solving. *IEEE Transactions on Systems, Man, and Cybernetics* 21(6): 1347–1363.

Lesser, V. R., and Corkill, D. D. 1981. Functionally Accurate, Cooperative Distributed Systems. *IEEE Transactions on Systems, Man, and Cybernetics* 11(1): 81–96.

Lesser, V. R.; Durfee, E. H.; and Corkill, D. D. 1989. Trends in Cooperative Distributed Problem Solving. *IEEE Transactions on Knowledge and Data Engineering* 1(1): 63–83.

Lewis, C. M., and Sycara, K. 1993. Reaching Informed Agreement in Multispecialist Cooperation. *Group Decision and Negotiation* 2(3): 279–300.

Liu, J., and Sycara, K. 1997. Coordination of Multiple Agents for Production Management. *Annals of Operations Research* 75:235–289.

Liu, J., and Sycara, K. 1995a. Emergent Constraint Satisfaction through Multiagent Coordinated Interaction. In *From Reaction to Cognition,* eds. C. Castelfranchi and J.-P. Muller, 107–121. Lecture Notes in Artificial Intelligence. New York: Springer-Verlag.

Liu, J., and Sycara, K. P. 1995b. Exploiting Problem Structure for Distributed Constraint Optimization. In *Proceedings of the First International Conference on Multiagent Systems,* 246–253. Menlo Park, Calif.: AAAI Press.

Liu, J., and Sycara, K. 1994. Distributed Problem Solving through Coordination in a Society of Agents. In *Proceedings of the Thirteenth International Workshop on Distributed Artificial Intelligence,* 190–206. Menlo Park, Calif.: AAAI Press.

Liu, J., and Sycara, K. P. 1993. Collective Problem Solving through Coordinated Reaction. In Proceedings of the IEEE International Conference on Evolutionary Computation, 575–578. Washington, D.C.: Institute of Electrical and Electronics Engineers.

Ljunberg, M., and Lucas, A. 1992. The OASIS Air-Traffic Management System. In Proceedings of the Second Pacific Rim International Conference on AI (PRICAI-92), 15–18 September, Seoul, Korea.

Loui, R. 1987. Defeat among Arguments: A System of Defeasible Inference. *Computational Intelligence* 3(1): 100–106.

Maes, P. 1994. Agents That Reduce Work and Information Overload. *Communications of the ACM* 37(7): 30–40.

Maes, P. 1990. *Designing Autonomous Agents.* Cambridge, Mass.: MIT Press.

Mason, C., and Johnson, R. 1989. DATMS: A Framework for Distributed Assumption-Based Reasoning. In *Distributed Artificial Intelligence, Volume 2,* eds. M. Huhns and L. Gasser, 293–318. San Francisco, Calif.: Morgan Kaufmann.

Minsky, M. 1986. *The Society of Mind.* New York: Simon and Schuster.

Müllen, T., and Wellman, M. P. 1996. Some Issues in the Design of Market-Oriented Agents. In *Intelligent Agents II,* eds. M. Wooldridge, J. P. Müller, and M. Tambe, 283–298. Lecture Notes in Artificial Intelligence 1037. New York: Springer-Verlag.

Müller, J. P., and Pischel, M. 1994. Modeling Interacting Agents in Dynamic Environments. In *Proceedings of the Eleventh European Conference on Artificial Intelligence (ECAI-94),* 709–713. Chichester, U.K.: Wiley.

Myerson, R. B. 1989. Credible Negotiation Statements and Coherent Plans. *Journal of Economic Theory* 48 (2): 264–303.

Newell, A. 1990. *Unified Theories of Cognition.* Boston: Harvard University Press.

O'Hare, G., and Jennings, N. 1996. *Foundations of Distributed Artificial Intelligence.* New York: Wiley.

Parsons, S., and Jennings, N. R. 1996. Negotiation through Argumentation—A Preliminary Report. In *Proceedings of the Second International Conference on Multiagent Systems,* 267–274. Menlo Park, Calif.: AAAI Press.

Parunak, V. 1987. Manufacturing Experience with the Contract Net. In *Distributed Artificial Intelligence, Volume 1,* ed. M. Huhns, 285–310. London: Pitman.

Rao, A. S., and George, M. P. 1991. Toward a Formal Theory of Deliberation and Its Role in the Formation of Intentions, Technical report, Australian Artificial Intelligence Institute, Victoria, Australia.

Rick, C., and Sidner, C. 1997. COLLAGEN: When Agents Collaborate with People. In Proceedings of the First International Conference on Autonomous Agents (Agents '97), 284–291. New York: Association of Computing Machinery.

Rosenschein, J. S., and Genesereth, M. R. 1985. Deals among Rational Agents. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI-85), 91–99. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Rosenschein, J. S., and Zlotkin, G. 1994. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers.* Cambridge, Mass.: MIT Press.

Sandholm, T. 1993. An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations. In Proceedings of the Eleventh National Conference on Artificial Intelligence, 256–262. Menlo Park, Calif.: American Association for Artificial Intelligence.

Sandholm, T., and Lesser, V. 1995. Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Protocol. In *Proceedings of the Second International Conference on Multiagent Systems,* 328–335. Menlo Park, Calif.: AAAI Press.

Schwuttke, U. M., and Quan, A. G. 1993. Enhancing Performance of Cooperating Agents in Real-Time Diagnostic Systems. In Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93), 332–337. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Sen, S., ed. 1996. Working Notes of the 1996 AAAI Symposium on Adaptation, Coevolution and Learning in Multiagent Systems. Menlo Park, Calif.: American Association for Artificial Intelligence.

Shehory, O.; Jha, S.; and Sycara, K. 1997. Multiagent Coordination through Coalition Formation. In *Agents IV, Agent Theories, Architectures, and Languages,* eds. M. Singh, A. Rao, and M. Wooldridge, 143–154. Lecture Notes in Artificial Intelligence. New York: Springer-Verlag.

Shoham, Y. 1993. Agent-Oriented Programming. *Artificial Intelligence* 60(1): 51–92.

Simon, H. 1957. *Models of Man: Social and Rational—Mathematical Essays on Rational Human Behavior in a Social Setting.* New York: Wiley.

Smith, I., and Cohen, P. 1996. Toward Semantics for an Agent Communication Language Based on Speech Acts. In Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-96). Menlo Park, Calif.: American Association for Artificial Intelligence.

Steeb, R.; Cammarata, S.; Hayes-Roth, F.; Thorndyke, P.; and Wesson, R. 1988. Distributed Intelligence for Air Fleet Control. In *Readings in Distributed Artificial Intelligence,* eds. A. H. Bond and L. Gasser, 90–101. San Francisco, Calif.: Morgan Kaufmann.

Stone, P., and Veloso, M. 1997. Multiagent Systems: A Survey from the Machine-Learning Perspective, Technical report, CMU-CS-97-193, School of Computer Science, Carnegie Mellon University.

Sycara, K. 1997. Using Option Pricing to Value Commitment Flexibility in Multiagent Systems, Technical report, CMU-CS-TR-97-169, School of Computer Science, Carnegie Mellon University.

Sycara, K. 1991. Problem Restructuring in Negotiation. *Management Science* 37(10): 1248–1268.

Sycara, K. 1990a. Negotiation Planning: An AI Approach. *European Journal of Operational Research* 46(2): 216–234.

Sycara, K. 1990b. Persuasive Argumentation in Negotiation. *Theory and Decisions* 28:203–242.

Sycara, K. P. 1988. Resolving Goal Conflicts via Negotiation. In Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88). Menlo Park, Calif.: American Association for Artificial Intelligence.

Sycara, K. 1987. Resolving Adversarial Conflicts: An Approach to Integrating Case-Based and Analytic Methods. Ph.D. dissertation, School of Information and Computer Science, Georgia Institute of Technology.

Sycara, K.; Decker, K.; and Zeng, D. 1998. Intelligent Agents in Portfolio Management. In *Agent Technology: Foundations, Applications, and Markets,* eds. M. Wooldridge and N. R. Jennings, 267–283. Berlin: Springer.

Sycara, K.; Roth, S.; Sadeh, N.; and Fox, M. 1991. Distributed Constrained Heuristic Search. *IEEE Transactions on System, Man, and Cybernetics* 21(6): 1446–1461.

Sycara, K.; Decker, K.; Pannu, A.; Williamson, M.; and Zeng, D. 1996. Distributed Intelligent Agents. *IEEE Expert* 11(6): 36–46.

Tambe, M. 1997. Toward Flexible Teamwork. *Journal of Artificial Intelligence Research* 7(1): 83–124.

Thomas, J., and Sycara, K. 1998. Stability and Heterogeneity in Multiagent Systems. In *Proceedings of the Third International Conference on Multiagent Systems.* Menlo Park, Calif.: AAAI Press.

Travers, M. 1988. Animal Construction Kit. In *Artificial Life,* ed. C. G. Langton, 421–442. Reading, Mass.: Addison-Wesley.

Van Dyke Parunak, H. 1987. Manufacturing Experience with the Contract Net. In *Distributed Artificial Intelligence,* ed. M. Huhns, 285–310. London: Pitman.

Veloso, M.; Stone, P.; Han, K.; and Achim, S. 1997. CMUNITED: A Team of Robotic Soccer Agents Collaborating in an Adversarial Environment. In *Proceedings of the First International Workshop on ROBOCUP.* San Francisco, Calif.: Morgan Kaufmann.

Wang, H., and Wang, C. 1997. Intelligent Agents in the Nuclear Industry. *IEEE Computer* 30(11): 28–34.

Weihmayer, R., and Velthuijsen, H. 1994. Application of Distributed AI and Cooperative Problem Solving to Telecommunications. In *AI Approaches to Telecommunications and Network Management,* eds. J. Liebowitz and D. Prereau. Amsterdam, The Netherlands: IOS Press.

Williamson, M.; Decker, K.; and Sycara, K. 1996. Unified Information and Control in Hierarchical Task Networks. In Proceedings of the AAAI-96 Workshop on Theories of Planning, Action, and Control. Menlo Park, Calif.: AAAI Press.

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

Zeng, D., and Sycara, K. 1998. Bayesian Learning in Negotiation. *International Journal of Human-Computer Studies* 48:125–141.

Zeng, D., and Sycara, K. 1997. Benefits of Learning in Negotiation. In Proceedings of the National Conference on Artificial Intelligence (AAAI-97), 36–41. Menlo Park, Calif.: American Association for Artificial Intelligence.

Zlotkin, G., and Rosenschein, J. 1991. Cooperation and Conflict Resolution via Negotiation among Autonomous Agents in Noncooperative Domains. *IEEE Transactions on Systems, Man, and Cybernetics* (Special Issue on Distributed Artificial Intelligence) 21(6): 1317–1324.

Katia P. Sycara is an associate research professor in the School of Computer Science at Carnegie Mellon University. She is also the director of the Advanced Information Technology Laboratory. She is directing-conducting research and developing technology in multiagent systems to support the decision making of multiple decision makers. For the past four years, she has led research in the development of distributed collaborating intelligent agents that interact with their users, other agents, and distributed information resources to assist their users in information finding and filtering and the planning and execution of various tasks.