MASSACHUSETTS INSTITUTE OF TECHNOLOGY

ARTIFICIAL INTELLIGENCE LABORATORY

Working Paper 228 January 1982

REPORT ON THE SECOND WORKSHOP ON DISTRIBUTED AI

edited by Randy Davis
MIT AI Lab

Abstract

On June 24, 1981 twenty-five participants from organizations around the country gathered in MIT's Endicott House for the Second Annual Workshop on Distributed AI. The three-day workshop was designed as an informal meeting, centered mainly around brief research reports presented by each group, along with an invited talk. In keeping with the spirit of the meeting, this report was prepared as a distributed document, with each speaker contributing a summary of his remarks.

Acknowledgments

A.I. Laboratory Working Papers are produced for internal circulation, and may contain information that is, for example, too preliminary or too detailed for formal publication. It is not intended that they should be considered papers to which reference can be made in the literature.

INTRODUCTION

This introduction tries to provide an overview of the workshop by identifying a few of the common threads that emerged from the discussions. The summaries follow, along with a list of attendees.[1]

Themes

Despite the diversity of approaches and application domains, a number of issues appeared to be central to many of the efforts. As in last year's meeting, the motivation to distributed problem solving often arose from characteristics of the problem under study. Distribution appears to be appropriate in the problems with multiple, active agents (robots, airplanes), natural geographic decompositions and high data rates (sensor nets), natural functional decomposition (offices), or the desire to distribute control (organizations). Distributed problem solvers may also be a source of power, since it may be easier to have twenty machines cooperate than it is to build one machine twenty, or even ten, times as fast.

But distributing problem solving also brings with it a host of problems. Once we distribute control, for example, it becomes less clear how we can ensure coherent, cooperative behavior. Once we distribute responsibility for various parts of a problem, each problem solver has only incomplete knowledge about the world, and those incomplete views may become inconsistent.

This year's gathering also emphasized the broad range of scales envisioned for the number and size of the computational elements used. The Rochester group, for example, impressed by the extensive parallelism evident at the neuronal level, discussed massive parallelism that would extend eventually to the scale of billions of very simple units. They described a connectionist model of computation, proposing it as an alternative to the sequential model associated with the traditional information processing view. Work at RAND and MIT, on the other hand, impressed by teamwork among groups of individuals, focused on cooperation on the scale of a few dozen large-scale problem solvers. Their work focused on the identification and collection of principles for cooperation, normative rules for behavior that produce cooperative action.

The talks addressed concerns ranging from formal theory to empirical measurements on working systems. Reports from SRI, for example, addressed issues in the theory of multi-agent planning, exploring a range of approaches considering how one agent can reason about the knowledge, abilities, and actions of another. Two reports from MIT described empirical results on the use of parallelism and distribution to reduce the computational complexity of certain problems. Work at U. mass. at Amherst has focused on the construction of a testbed for evaluating a number of parameters in the design of a distributed interpretation system.

Finally, a predictably broad range of application domains is under study. Domains include signal processing (DREA, AI&DS), air traffic control (RAND), robot planning (SRI, MIT), management (CMU), office work (Xerox, MIT), and natural language (Rochester, Toronto).

---

1. A report on the first workshop can be found in the Sigart Newsletter No.73, October 1980.

Connectionist Models and Their Properties

J.A. Feldman and D.H. Ballard
Computer Science Department
University of Rochester - Rochester, N.Y. 14627

Much of the progress in the fields constituting cognitive science has been based upon the use of concrete information processing models (IPM), almost exclusively patterned after conventional sequential computers. There are several reasons for trying to extend IPM to cases where the computations are carried out by a massively parallel computational engine with perhaps billions of active units. The current interest in massively parallel models can be motivated from four different perspectives: anatomy, computational complexity, technology, and the role of formal languages in science. It is the later which is of primary concern here. We focus upon a particular formalism, connectionist models (CM), which is based explicitly on an abstraction of our current understanding of the information processing properties of neurons.

The critical resource that is most obvious is time. Neurons whose basic computational speed is a few milliseconds must be made to account for complex behaviors that are carried out in a few hundred milliseconds [Posner]. This means that *entire complex behaviors are carried out in less than a hundred time steps.* Current AI and simulation programs require millions of time steps. It may appear that the problem posed here is inherently unsolvable and that we have made an error in our formulation. But recent results in computational complexity theory [JaJa] suggest that networks of active computing elements can carry out at least simple computations in the required time range. We have developed fast solutions to a variety of relevant computing problems. These solutions involve using massive numbers of units and connections, and we also address the questions of limitations on these resources.

The most important reason for a serious concern in cognitive science for CM is that they might lead to better science. It is obvious that the choice of technical language that is used for expressing hypotheses has a profound influence on the form in which theories are formulated and experiments undertaken. Artificial intelligence and cognitive sciences have made great progress by employing models based on conventional digital computers as theories of intelligent behavior. But a number of crucial phenomena such as associative memory, priming, perceptual rivalry, and the remarkable recovery ability of animals have not yielded to this treatment. A major goal of this work is to lay a foundation for the systematic use of massively parallel connectionist models in the cognitive sciences, even where these are not yet reducible to physiology or silicon.

References

[1] Ja'Ja', J. and J. Simon, "Parallel algorithms in graph theory: Planarity testing," CS 80-14, Computer Science Dept., Pennsylvania State University, June 1980.

[2] Posner, M.I. *Chronometric Explorations of Mind.* Hillsdale, N.J.: Lawrence Erlbaum Associates, Publishers 1978.

The Rochester Discourse Comprehension Project

James F. Allen and Steven L. Small
Computer Science Department
University of Rochester - Rochester, N.Y. 14627

We are conducting research into highly interactive man-machine dialogues that concern some task that the machine is assisting the user perform, or some complex behavior that the machine is attempting to explain to the user. In these domains there is a well defined set of goals that is pursued by the user throughout the dialogue. "Highly interactive" refers to the fact that these dialogues are not restricted to simple turn-taking by the participants, as is common in current question-answering and discourse systems. Currently involved in the project are James Allen, Steven Small, Gary Cottrell, Alan Frisch, Diane Litman, Lokendra Shastri, and Marc Vilain.

System Architecture

The overall system architecture can be motivated by the observation that to analyze a conversation one must consider the goals of the participants at many different levels. The levels most relevant to task-oriented dialogue include the communicative level, which includes goals such as "get someone's attention," "deny a previous point," "change topic," etc., and the task level, which includes goals such as "get a tape mounted," "find out Sue's address," "assemble a water pump," "board a train."

The system separates these levels into distinct subsystems which run in parallel and communicate via message passing. Each module may access information from a centralized knowledge base which maintains facts and provides a set of automatic inferences (e.g. inheritance as found in semantic networks). This module also provides the mechanism to index facts by belief and other propositional attitudes, as well as providing temporal reasoning [Allen, 1981].

The system will involve two levels of goal recognition. The first level takes a literal analysis of the input and attempt to recognize the communicative goal being pursued (as in [Allen, 1979]). Once this goal is recognized, it can be used to identify the operations at the task level, and the task level goals can be recognized. For the purposes of this discussion, the levels appear to be manipulated in sequence. In fact, however, the recognition processes will be working at all levels of the system simultaneously. Thus, the level which first finds a likely analysis will be able to assist the analysis at the other levels.

The organization of the system is shown in Figure 1.

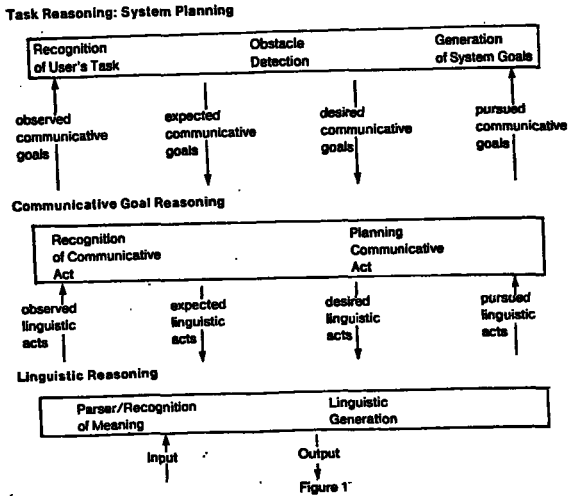

Figure 1

An important aspect of the Rochester Project is the construction of a Word Expert Parsing system that operates in the full environment of parallel inference modules cooperatively understanding discourse. Such an analyzer must perform a large number of functions, including: (1) disambiguating word senses; (2) finding referents for definite noun phrases and pronouns; (3) understanding idioms, noun pairs, and other idiosyncratic lexical strings; (4) understanding metaphor; and (5) learning new words and word senses. The ability of man or machine successfully to perform these cognitive tasks requires significant context of discourse and logical knowledge, and the ability to use it. Building a computer program to do all of these things (at least partially) requires some kind of analogous computational context.

There is a reason why we believe that the architecture of the system must be distributed, both at the levels of parsing, goal recognition, planning, and other major subsystems, and within those levels. (Note that the block diagram in the figure does not show the levels of distributed processing internal to these systems.) A fundamental underlying perspective of the Rochester Project is that a discourse comprehension system must to a significant extent focus on the interactions among the various cooperating subsystems, i.e., what and how they communicate to each other. That is not to say that we are not interested in the development of perspicuous subsystems; we do believe, however, that the interactions among these systems must be considered of equal import.

To this extent, we are developing a set of high-level conventions for implementing an interaction-based distributed comprehension system. Individual components are being designed within this framework, with their external sources of knowledge kept clearly in mind at all times.

References

[1] Allen, J.F., "A Plan-Based Approach to Speech Act Recognition," Ph.D. thesis, Computer Science Department, University of Toronto, 1979.

[2] Allen, J.F. "Maintaining Knowledge about Temporal Intervals," TR 86, Computer Science Department, University of Rochester, 1981.

[3] Small, S.L. "Viewing Word Expert Parsing as Linguistic Theory," *Proceedings of the 7th IJCAI,* Vancouver, B.C., August 1981.

[4] Small, S.L. "Demon Timeouts: Limiting the Life Span of Spontaneous Computations in Cognitive Models," 3rd Conference, Cognitive Science Society, Berkeley, Ca., August 1981.

Teamwork in Multi-Agent Planning; Distribution as an Approach to Complexity

Randy Davis, Dan Brotsky, Judy Zinnikas
Artificial Intelligence Laboratory
Massachusetts Institute of Technology - Cambridge, Ma. 02139

An underlying assumption in traditional work on planning is the uniqueness of the agent generating the plans: all plan generation activity is done by a single agent. Recently, however, attention has been given to the problems inherent in having multiple agents formulate and execute plans. Our efforts [1,2] in this direction have been focused on the planning act itself, and have considered the nature of and style of planning appropriate to a world in which more than one active agent is present. We have attempted to specify the characteristics of such a world that make it different from the traditional single-agent world, and have considered what impact these characteristics have on the attempt to generate plans.

We find, for example, that the presence of multiple agents creates an interesting form of uncertainty: things will change in the world without an agent being able to predict them exactly. As a result, plans should be *robust*, i.e., relatively insensitive to changes in the world produced by the other agents. The plans should also be *cautious*, i.e., they should reduce the number of opportunities for other agents' actions to interfere. At the same time, the group of agents is assumed to be a team working together and hence needs to develop plans that are *cooperative*.

This represents a departure from traditional models that focus on generating very good plans in a completely known and predictable world. Here we may be more interested in a plan that is good and very flexible in the face of contingencies, rather than one that is very good but relatively inflexible. It also represents a change from earlier approaches to dealing with uncertainty in planning. Previous approaches focused on uncertainty arising from events characterizable only probabilistically (e.g., the probability that a block would fall out of the hand while being moved), while here we are dealing with goal-seeking agents and can rely on inferences about their plans to generate a restricted set of contingencies we want to deal with.

In the most general terms, our concern is with identifying and accumulating *principles for teamwork*. What is it that allows a number of individuals to formulate plans for the most part independently and yet have those plans mesh well together? Borrowing from a range of disciplines, we have identified principles like slack resources, behaving predictably, generating simple plans, and using knowledge about the task domain to make predictions about characteristics of the plans other agents will generate. Our work involves constructing a planning system that incorporates such principles.

We have also been exploring [3] how distribution and cooperation can reduce the computational complexity of some algorithms. Like the work in [4], we consider how communication between processors can produce significant speedups. We have taken a very simple problem and begun to study in some detail the effect of communication. We propose a number of different simple architectures and analyze the behavior of each, attempting to understand how communication affects the complexity of the calculation, and attempting to characterize the class of algorithms for which such speedup is possible.

References

[1] Davis R, Smith R G, Negotiation as a metaphor for distributed problem solving, *Artificial*

Intelligence, to appear, 1982.

[2] Davis R, A model for planning in a multi-agent environment: steps toward principles for teamwork, MIT AI Lab Working Paper 217, October 1981.

[3] Brotsky D C, Davis R, Complexity reduction through cooperation, in preparation.

[4] Kornfeld W, Combinatorially implosive algorithms, *CACM*, to appear.

CNET: Simulation of Distributed Problem Solving

Reid G. Smith
Defence Research Establishment Atlantic
Dartmouth, NS, Canada

CNET is a medium for writing programs for distributed problem solvers. It is based on a system of INTERLISP [Teitelman, 1978] functions that simulates the operation of a distributed problem solver whose nodes communicate via the *contract net protocol* [Smith, 1980a].

Distributed problem solving is currently at a stage where basic options for control and knowledge distribution have been identified, but little experience has been collected and few experiments have been performed. Experimentation is required to provide detailed answers to questions about what task-specific information to communicate, how to combine conflicting results from different nodes and coordinate node activities to maintain global coherence, and how to manage the tradeoff between communication and computation.

Given the experimentation task at hand, a number of approaches are possible. The approach taken in CNET has been to construct a system that could be incorporated into a distributed problem solver. This allows us to gain experience in actual construction of such problem solvers, and to acquire a feel for the practical utility and difficulties associated with the various approaches to control (e.g., task-sharing and result- sharing). This approach should produce results that are complementary to those produced with the parametric model of the University of Massachusetts group [Lesser, 1981].

CNET is composed of a number of modules: *Simulation, Common Internode Language, Knowledge Representation*, and *User Interface*. The Simulation module contains the software necessary to maintain quasi-parallelism. [The task execution procedure associated with each contract is set up as a *generator* using the INTERLISP spaghetti stack.] The Common Internode Language module contains the parser, grammars, and related functions for handling the contents of message slots.

The Knowledge Representation module is based on the **Unit Package** [Stefik, 1980], [Smith, 1980b], with its access functions and interactive editor. Most of the work of the past year has been spent on this module in an attempt to provide a knowledge representation language that is powerful enough to be useful in real problems.

The User Interface module contains functions for user adjustment of simulation parameters (e.g., number of processor nodes, message delays, coupling factor). It also contains the user-callable functions for message-passing, task distribution, reporting, and so on. The programming language available to a user for problem solving with CNET is thus INTERLISP augmented by these functions. The User Interface module functions encapsulate the constructs available to the user for the distributed aspects of problem solving.

CNET is being tested mainly in the domain of distributed sensor data interpretation. This domain is appropriate because: (i) it has a naturally distributed character, (ii) it is a domain in which both task-sharing and result-sharing are at least appropriate if not essential, and (iii) error-prone and conflicting results are a regular occurrence.

References

[1] [Lesser, 1981] V.R. Lesser, P. Bates, R. Brooks, D. Corkill, E. Hudlicka, L. Lefkowitz, R. Mukunda, J. Pavlin, S. Reed, J. Wileden, *A High-Level Simulation Testbed for Cooperative Distributed Problem Solving*. COINS TR 81-16, Department of Computer and Information Science, University of Massachusetts, June 1981.

[2] [Smith, 1980a] R.G. Smith, The Contract Net Protocol: High-Level Communication And Control In A Distributed Problem Solver. *IEEE Trans. on Computers*, Vol. C-29, No. 12, December 1980, pp. 1104-1113.

[3] [Smith, 1980b] R.G. Smith and P.E. Friedland, *Unit Package User's Guide*. DREA TM 80/L, Defence Research Establishment Atlantic, Dartmouth, NS, Canada, December 1980. (Also published as HPP-80-28, Heuristic Programming Project, Stanford University.)

[4] [Stefik, 1980] M.J. Stefik, *Planning With Constraints*. STAN-CS-80-784 (HPP-80-2), Dept. of Computer Science, Stanford University, January 1980.

[5] [Teitelman, 1978] W. Teitelman, *INTERLISP Reference Manual*. Xerox Palo Alto Research Center, Palo Alto, Ca., October 1978.

Distributed Interpretation Testbed

V.R. Lesser and D.D. Corkill
Department of Computer Science
University of Massachusetts at Amherst - Amherst, Ma. 01002

During the last two years, the major focus of our efforts has been the building of a distributed interpretation simulation testbed [Lesser, et. al. 1981]. The purpose of this testbed is to provide a research tool for empirically evaluating alternative designs for cooperative distributed problem-solving systems. An initial version of the testbed has been completed and is now being used on a regular basis.

The interpretation task we have chosen for the testbed is distributed traffic monitoring. The problem is to identify, locate, and track patterns of vehicles moving in a two-dimensional space. As a result of normal operation, vehicles generate acoustic signals, some of which are received by sensors reporting to a monitoring system. The monitoring system produces and maintains a map of traffic in the environment. (This task is similar to one used by Smith [1980].) Both vehicle and sensor characteristics are variable, permitting control of the spatial distribution of ambiguity and error in the task input data. Node configurations and communication channel characteristics can also be independently varied in this simulated system.

The testbed simulates a network of nodes, each of which is an architecturally-complete Hearsay-II system [Lesser and Erman, 1980]. Interprocessor communication is naturally included in the architecture by adding knowledge sources on each blackboard level to send and receive messages. Our simulation model permits arbitrary node-node, node-sensor connectivity and arbitrary distributions of task processing capability across the nodes in the system.

In order to permit more explicit forms of control among nodes in the system, we have integrated goal-directed control into the data-directed control structure of the Hearsay-II architecture. This permits nodes to affect the processing of other nodes not only through the transmission of hypotheses, but also through transmission of goals, which are requests for the creation of hypotheses with certain attributes. This integrated control framework also improves the scheduling decisions made within each node [Corkill and Lesser, 1981].

Incorporated into the testbed are capabilities for varying the accuracy of individual knowledge sources. This is accomplished through the use of an oracle that can compare the developing interpretations with the interpretation that would be produced if the system had perfect knowledge. These capabilities permit the study of how different control and communication policies perform under varying distributions of uncertainty and error in the intermediate states of processing. The oracle also permits development of measures for intermediate states of processing in the network and has greatly reduced the knowledge-engineering effort [Lesser, Pavlin, and Reed, 1980].

We are currently running a set of experiments in the testbed with a simple scenario, comparing the performance of a centralized version, a 4-node system in which nodes cooperate through the communication of high level hypotheses, and a 5-node hierarchical system which is similar to the 4-node system except that, instead of communicating among themselves, nodes send their hypotheses to a fifth node that constructs the answer. These experiments have already demonstrated that realistic, alternative system behaviors are obtained by varying the accuracy of oracle-based knowledge sources.

References

[1] Corkill, D. D., and Lesser, V.R., "A Goal-Directed Hearsay-II Architecture: Unifying Data-Directed and Goal-Directed Control," Technical Report 81-15, Department of Computer and Information Science, University of Massachusetts, Amherst, Ma. June 1981.

[2] Lesser, V.R., et al, "High-Level Simulation Testbed for Cooperative Distributed Problem-Solving," Technical Report 81-16, Department of Computer and Information Science, University of Massachusetts, Amherst, Ma. June 1981.

[3] Lesser, V.R., and Corkill, D.D., "Functionally Accurate, Cooperative Distributed Systems," *IEEE Transactions on Systems, Man and Cybernetics,* Vol. SMC-11, No. 1, pp. 81-96, January 1981.

[4] Lesser V.R., and Erman, L.D., "Distributed Interpretation: A Model and an Experiment," *IEEE Transactions on Computer,*Vol. C-29, 12, pp. 1144-1162, December 1980.

[5] Lesser, V.R., Pavlin, J., and Reed, S., "Quantifying and Simulating the Behavior of Knowledge-Based Interpretation Systems," *Proceedings of the First Annual National Conference on Artificial Intelligence,* pp. 111-115, August 1980.

[6] Smith, R.G., "A Framework for Problem Solving in a Distributed Processing Environment," *IEEE Transactions on Computers, Special Issue on Distributed Processing Systems,* Vol. C-29, No. 12, pp. 1104-1113, December 1980.

Distributed Problem Solving in Air Traffic Control

Perry Thorndyke, David McArthur,
Stephanie Cammarata, Randy Steeb

Rand Corporation
Santa Monica, Ca. 90405

DAI is concerned with problem-solving situations in which several agents cooperate to achieve a common set of objectives. These agents may have different and incomplete knowledge bases, different expertise, and different amounts of computational resources available. In many multi-agent problem-solving contexts, performance is poor because agents often foil, not support, the actions of others. Therefore, one goal of DAI research should be the development of a theory of cooperation and teamwork. Such a theory should dictate: (i) how to organize multi-agent systems--how many agents should cooperate and in what organizational structure; (ii) how the global problem should be distributed among agents to best utilize their local expertise and resource constraints; (iii) when and how agents should communicate their intentions to others who may have incomplete knowledge of those intentions.

We have approached the theory of cooperation as an empirical investigation. In particular, we aim to develop normative rules of cooperation by examining several distinct cooperative strategies in a variety of problem situations. We characterize these various problem contexts in terms of a set of environmental attributes. We then evaluate the effectiveness of each cooperation strategy in these situations while holding constant the expertise available to each agent.

To accomplish this design-and-test cycle in a systematic fashion, we are developing a flexible experimental environment. This environment makes it simple to construct many specific distributed problem solvers. Users will be able to rapidly implement new designs, including the selection of a distributed organization or architecture and the specific cooperative regime to be followed. The resulting distributed system can then be tested on problem-solving tasks with a variety of environmental characteristics.

We are conducting these DAI experiments using the domain of air traffic control (ATC). In the air traffic control simulation embedded in our experimental environment, multiple problem-solvers guide incoming aircraft through a three-dimensional airspace (14X24 miles X 9000 vertical feet). Incoming aircraft include overflights, airport arrivals, and airport departures. Aircraft flight plans must be developed and executed so that all aircraft maintain at least 3 miles horizontal or 1000 feet vertical separation. This task is challenging because the airspace can become quite dense, requiring many distinct planning decisions in a short period of time. In our simulation, density is a parameterized environmental attribute, so we can investigate a wide variety of problem situations.

We feel ATC is a good domain in which to investigate distributed problem solving for several reasons. First, ATC is a time stressed problem, and, especially when the airspace is crowded, formulating conflict-free plans sufficiently rapidly may be impossible for a single processor. Second, many architectures are possible for the distribution of planning effort, thus facilitating the investigation of a theory of cooperation that specifies optimal architectures for particular environments. For example, agents may be assigned to individual aircraft (an object-centered architecture), sectors of the airspace (space-centered), or levels of planning responsibility (hierarchical) [1]. In addition, a variety of control regimes might be employed to resolve potential conflicts between approaching aircraft [2].

To date we have developed an initial experimental environment, implemented in Interlisp. Using this environment, we have investigated two collaboration regimes within the object-centered

architecture: an autonomous variant and a cooperative variant. The versions differ with respect to the extent to which agents can communicate with one another. In the autonomous planning variant, the communication modules of each agent have been "turned off", and agents must rely on sensing and inferencing capabilities to determine the locations, and intentions of other aircraft. Each aircraft must rely on its own planning capability to avoid conflicts with other aircraft in the airspace, much like cars on a freeway maneuver autonomously. In the object-centered cooperative architecture, aircraft can communicate plans, intentions, and locations, and can jointly plan to avoid conflicts. While providing more planning options, this architecture forces aircraft to depend on rapid and error-free communications during cooperation.

We have conducted experiments contrasting the performance of these architectures under different conditions of airspace density. When air traffic is low or moderate, the autonomous planning produces conflict-free plans as frequently as cooperative planning. In these cases, there is sufficient maneuverability in the airspace for aircraft to avoid conflicts through their own planning efforts. However, when the airspace has high traffic density, the cooperative version resolves significantly more aircraft conflicts than the autonomous version. In this case, the airspace density constrains the set of solutions to conflicts, and aircraft must cooperative to coordinate their adoption of mutually satisfactory plans.

We plan to extend our efforts to develop a theory of cooperation in several directions. First, we will design and test several more distributed problem solvers in the ATC domain. The dimensions we will vary include (i) the kinds of cooperation and negotiation strategies used to resolve shared conflicts; (ii) the costs and benefits of using resources critical to cooperation, including, for example, delay, reliability, and bandwidth of communications; and (iii) the types of distributed architectures. Second, we intend to improve our experimental environment so that it is easier to design various problem solvers. To do so, we are designing a new problem-solving system in ROSIE, an English-like rule-based language developed at Rand, that will facilitate the discovery of and experimentation with new cooperation heuristics.

References

[1] R. Steeb, S. Cammarata, F. Hayes-Roth, and R. Wesson, Distributed intelligence for air fleet control. WD-839-ARPA, The Rand Corporation, December, 1980.

[2] P. Thorndyke, D. McArthur, and S. Cammarata, "Autopilot: A distributed planner for air fleet control," *Proceedings of the Seventh IJCAI*, pp. 171-177, Vancouver, B.C. Canada, August, 1981.

Omega and its Presenter

Carl Hewitt
L.C.S. - A.I. Lab.
Massachusetts Institute of Technology
Cambridge, Ma. 02139

At the workshop I discussed some recent work by Gerald Barber and Gene Ciccarelli on Omega and its current graphical interface called the "Presenter". Their work builds on the doctoral research of Bill Kornfeld in developing the Ether system as well as an implementation of a preliminary version of Omega done previously by Attardi, Simi, and Barber.

I discussed the description system Omega and some recent work by Gerald Barber on its implementation on the personal computer developed by the MIT Artificial Intelligence Laboratory. I put forth some hypotheses about aspects in which Omega has improved over previous systems such as KRL and FRL. An important improvement in Barber's implementation of Omega is the integration of pattern directed invocation (Sprites) into Omega's inheritance network. Some examples were presented elucidating the structure of the implementation that showed subtle relationships between the declarative statements and imperative actions of the implementation.

The Omega Presenter is a graphical interface to Omega, a tool developed by Gene Ciccarelli for viewing and adding to Omega networks. I showed some of its design and operational characteristics, emphasizing how they aid in using Omega. In particular I discussed some recent work by Ciccarelli on: the need for different ways to view network, control of context, control of the amount of detail seen, and showing structure. We think of Omega as "having only one kind of link"; this simplifies the design of a general graphical interface.

References

[1] Attardi, G. and Simi, M. "Semantics of Inheritance and Attributions to the Description System Omega", *Proceedings of the Seventh IJCAI*, Vancouver, B.C. Canada, August, 1981.

[2] Barber G.R. "Embedding Knowledge in a Workstation", *Proceedings of the INRIA International Workshop on Office Information Systems*, October 1981.

[3] Barber, G.R. "Reasoning about Change in Knowledgeable Office Systems", *First Annual National Conference on Artificial Intelligence*, August 1980.

[4] Hewitt, C., Attardi, G. & Simi, M. "Knowledge Embedding with a Description System", *M.I.T. A.I. Memo*, August 1980.

[5] Kornfeld, W.A. & Hewitt, C. "The Scientific Community Metaphor", *IEEE TRansactions on Systems, Man, and Cybernetics*, Vol. SMC-11, No. 1, January 1981.

The Ether Language

Bill Kornfeld
L.C.S. - A.I. Lab.
Massachusetts Institute of Technology
Cambridge, Ma. 02139

The Ether language, a language for parallel problem solving, was reported on at the first Workshop on Distributed AI. Since that time it has been considerably improved and extended. Two large systems have been implemented in it to test various features and explore the notions of parallel problem solving.

One of these is a program synthesis system that takes as input a description of the relationship between inputs and output of a Lisp function in a variant of first order predicate logic. The system is intended to synthesize a Lisp program from this description. The system was designed to test the notion of pursuing multiple implementation strategies in parallel with *opponent strategies*. The opponent strategies attempt to show that certain proposed implementation strategies cannot possibly work. If these succeed, work on the successfully refuted implementation strategy is halted.

The second system is one that solves cryptarithmetic puzzles like the ones in Newell and Simon's *Human Problem Solving*. This system is capable of pursuing several alternative hypotheses about assignments of letters to digits in parallel. The resources given to these various activities can be changed asynchronously with their running. Strategies for resource allocation were described. The relationship between *parallel search* and standard tree search algorithms was described.

Several implementation details were discussed. The most significant of these is the notion of *virtual collections of assertions*. The discrimination net common to implementations of pattern-directed invocation systems has been completely replaced by a scheme that compiles the assertions and data-driven procedures into much more efficient code. The form of this code is specified by the programmer for classes of assertion types and is suggested by their semantics. The technique significantly improves the flexibility and efficiency of this class of languages.

References

[1] Kornfeld, W.A., and Hewitt, C.E. "The scientific community metaphor," *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11, 1, January 1981.

[2] Kornfeld, W.A., "Using parallelism to implement a heuristic search," AI Memo 627, MIT, Artificial Intelligence laboratory, Cambridge, Ma., March 1981, also in *Proceedings of IJCAI-81*, Vancouver, B.C., August 1981.

The Intelligent Management System Project

The Robotics Institute
Carnegie-Mellon University - Pittsburgh, Pa. 15213

Personnel: Mark Fox, Tom Morton, Gary Strohm, Drew Mendler,
Joe Mattis, Steve Miller, Ari Vepsalainen, Robert Federking.

The Intelligent Management System (IMS) is a distributed AI system for managing organizations. Its long term goals include the sensing, modeling, management and analysis of organizations such as factories.

PHASE I (March 80 - Aug. 80) of the project was an analysis of the functional, user-interface, and system architecture requirements for the host factories. Three factories were analyzed: a turbine component production plant, a circuit board production plant, and a lamp production plant, all part of the Westinghouse Corporation. The major functions required include: interactive, real-time job-shop scheduling, manager accessible factory analysis such as simulation, task planning and monitoring, and process control and diagnosis. The user interface requirements centered around allowing managers to use the system. The interface must be accessible: natural language and discourse analysis; be accountable: explanation facility; amplify requests: plan how questions are to be answered or commands executed; and be adaptable: learn from user interactions the organization model changes. The architecture requires multiple processes. Each user must have a User Interface Process (UIP), each task a Task Interface Process (TIP), and each machine a Machine Interface Process (MIP). These processes must communicate information and cooperatively solve problems and carry out commands.

PHASE II (July 80 - April 81) was a design and implementation of selected functions. The first problem was to construct an organization modeling system to support the various functions. The SRL knowledge representation language is used to model organizations. One of its facilities allows users to define their own inheritance relations. An interactive job-scheduling system was designed and constructed. It is a constraint driven scheduling system, which used due dates, cost, operation and machine processing restrictions, and machine breakdowns as constraints. An interactive simulation system with color display was constructed, and uses the same model as the scheduling system; it allows run-time monitoring, analysis, and alteration of a factory model simulation.

PHASE III (March 81 - Dec. 81) of the project included a redesign of the currently implemented functions. The scheduling system is being altered to include more constraints, e.g., production goals, resource availability, forecasts, dynamic lotting, sub-assembly sequencing, and relax certain constraints (e.g., due date) when needed. The simulation system allows multi-level simulation, and model consistency checking. Work has begun on a natural language system and a discourse model for the UIP to support scheduling and model building. A capability system for restricting information and function access is also being built. The Organization Description Language (ODL), a language for describing distributed AI systems, is being altered and used to describe all modules, processes, channels, and ports in the system.

PHASE IV (Jan. 82 - ?) will include a UIP planning system which will reason about other processes, using their ODL descriptions, in order to do (distributed) problem-solving. The next version of the scheduling system will be implemented in Hearsay-II-like architecture to allow opportunistic use of constraints.

References

[1] (IMS) Fox M.S., (1981), "The Intelligent Management System: An Overview", Technical Report, Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, July 1981.

[2] (Simulation) Reddy Y.S., & M.S. Fox, (1981), "Knowledge-Based Simulation", Technical Report, Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, in preparation.

[3] (SRL) Fox M.S., (1979), "On Inheritance in Knowledge Representation", *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, Tokyo Japan.

[4] Fox M.S., (1981), "SRL: Schema Representation Language", Technical Report, Robotics Institute, Carnegie-Mellon University, Pittsburgh PA, in preparation.

[5] (ODL) Fox M.S., (1979), "Organization Structuring: Designing Large, Complex Software", Technical Report CMU-CS-79-155, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA.

[6] (Distributed Problem Solving) Fox M.S., (1981), "Reasoning With Incomplete Knowledge in a Resource-Limited Environment: Integrating Reasoning and Knowledge Acquisition", Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, BC, Canada, Aug. 1981.

A Commitment-Based Framework for Describing Informal Cooperative Work

Richard E. Fikes
Cognitive and Instructional Sciences Group
Xerox Palo Alto Research Center - Palo Alto, Ca. 94304

In this talk I presented a framework for describing cooperative work in domains such as an office where there is no agreed upon formal model of the tasks and functions to be done, nor of the procedures for doing them. Our goal in creating the framework was to provide a basis for the design of computer-based systems capable of performing and supporting work in such situations. We argue that standard descriptive techniques (e.g., flow charts) are inadequate for expressing the adaptability and variability that is observed in offices, and are fundamentally misleading as metaphors for understanding the skills and knowledge needed by computers or people to do the work.

The basic claim in our alternative framework is that an agent's work is defined in terms of making and fulfilling commitments to other agents. The tasks described in those commitments are merely agreed upon means for fulfilling the commitments, and the agents involved in the agreement decide in any given situation how and whether a given commitment has been fulfilled. Hence, methods and criteria for task completion are determined on a case-by-case basis by the agents involved through a process of negotiation.

We observe that in informal domains, determining the meaning of general task and procedure descriptions in specific situations is an important component of the work. We claim that the sole authorities for determining the meaning of those descriptions are the agents who have contracted to do the tasks or functions. Therefore, those descriptions and their interpretation evolve during the course of their use in continuing negotiations as situations and questions arise in which their meaning is unclear. These claims imply that for a given situation an agent using such descriptions must be capable of interpreting imprecise descriptions, determining effective methods for performing tasks, and negotiating with other agents to determine task requirements.

We also observe that in informal domains, there are no guarantees that a procedure will successfully accomplish its task. Those guarantees are lost because the procedure, its task, and the situations in which it will be used are imprecisely described. Hence, procedures in informal domains are only prototypes of methods for performing tasks. They suggest a way of decomposing a task into steps, and perhaps indicate how the task is typically performed, but they do not alleviate the need for problem solving in each specific situation to determine how to perform a task. We suggest ways of augmenting a procedure description to provide the information needed to do that problem solving. Those suggestions include providing a description of the goals to be achieved by each procedure step, the consumer of the results of each step, and a description of how the results are to be used by the consumer.

References

[1] Fikes, R. "A Commitment-Based Framework for Describing Informal Cooperative Work". Proceedings of the Cognitive Science Conference, Berkeley, Calif., August 1981.

[2] Fikes, R. "Automating the Problem Solving in Procedural Office Work". In Proceedings of the AFIPS Office Automation Conference, Houston, Texas, March 1981.

[3] Fikes, R., and Henderson, A. "On Supporting the Use of Procedures in Office Work". Proceedings of The First Annual National Conference on Artificial Intelligence, Stanford, Calif., August 1980.

Distributed Hypothesis Formation (DHF) in Distributed Sensor Networks (DSN)

R. P. Wishner, R. J. Drazovich, V. G. Rutenberg,
C. Y. Chong, J. Abram, F. X. Lanzinger
Advanced Information & Decision Systems (AI&DS)
Mountain View, Ca. 94040

This project involves developing techniques for formulating hypotheses about an observed situation in a distributed fashion. The project model involves a set of processors connected into (say) a packet switching network. Each node (processor) has its own sensors (e.g., acoustic, radar) and limited communications with its neighbors. The nodes have overlapping sensor coverage. The goal is to develop techniques so that, in a distributed manner, all nodes can obtain and maintain a global hypothesis about the overall observed situation. The research issues are (1) how does an individual node formulate hypotheses about the situation using as input his own sensor information and the summarized hypotheses provided by neighboring nodes. This can be viewed as a knowledge-based hypotheses formation problem that must integrate information from multiple sources in a time-evolving situation. (2) How can information best be propagated through the network? What information should be transmitted? What is the appropriate control structure (to avoid looping, etc.)?

Research is just underway (in July, 1981): Plans are to survey existing technology, design various control structures and hypothesis formation techniques, and develop software to test some of the techniques. Both artificial intelligence (knowledge based) approaches and more traditional techniques will be considered. A specific domain will be selected for use in this research. An air defense situation is one possibility currently being considered.

References

[1] Drazovich, R. J., Brooks, S., Payne, J. R., Lowerre, B., Foster, S., "Surveillance Integration Automation Project (SIAP): Technical Progress Report, November 1977 to January, 1979", Systems Control, Inc., Palo Alto, CA

[2] Reid, D. B., "An Algorithm for Tracking Multiple Targets," IEEE Trans. Auto. Control, AC24(6), 843:854, 1979.

Distributed Decision Making

R. P. Wishner, C. Y. Chong, J. Abram, R. J. Drazovich,
V. G. Rutenberg, B. P. McCune, J. S. Dean
Advanced Information & Decision Systems (AI&DS)
Mountain View, Ca. 94040

This research is an Air Force sponsored effort to develop a design methodology and technology for distributed decision making in the context of Air Force tactical command, control and communications (C-3) problems, in particular, situation assessment, planning, and control.

Distributed decision making is a complicated problem for which no single technical approach

is likely to be successful. As a result we will be using techniques from AI, control theory, as well as other disciplines. In addition to surveys of relevant areas, we hope to develop a top-down design methodology to allow us to decompose a decision problem into logical sets of tasks, map these tasks into candidate solution techniques involving subsets of different technologies, evaluate the design and iterate on additional modifications. Work on this project is just underway (in July, 1981).

Summary of DAI Research At SRI

Stan Rosenschein
SRI International
Menlo Park, Ca. 94025

The DAI group at SRI has been concerned primarily with theoretical issues that arise in attempting to extend classical AI planning methods to include multiple agents.

Our general assumptions can be described as follows: A robot agent inhabits an environment which contains, along with the usual inanimate objects, other similar agents capable of independent action. The state of the world is described in terms of the physical state of the inanimate objects and the cognitive state (beliefs, goals) of the other agents. To plan and act effectively in such a world, the robot must reason about the effects of its actions--including their effect on the cognitive state of other agents. For instance, it may be possible to achieve "physical" goals (e.g. moving a box) by achieving "cognitive" subgoals (e.g. getting another agent to want to help move it, say by asking him).

Most of our attention to date has gone to developing formalisms adequate for representing in detail the effects of actions on the cognitive state of other agents [1,2]. In the area of planning, we have been looking at ways of modifying the classical AI paradigms (e.g. STRIPS and situation calculus) to accommodate multiple agents [3]. Some of this work has led to new theoretical perspectives on planning in general [4]. In addition, we have been trying to find appropriate models of distributed action to serve as a theoretical foundation for DAI, supplanting the sequential, state-transformation model underlying such systems as STRIPS and requiring, perhaps, new planning techniques. One question we are asking, for instance, is whether the "parallelism" in Sacerdoti's procedural nets can be given a semantics that is useful for the needs of distributed planning.

Another aspect of our work at SRI involves studying alternative ways a planner might reason about how another agent (with particular beliefs and goals) will act. In certain instances the robot planner will want to simulate the planning mechanism of other robots in some detail. On other occasions, however, it may be useful for the robot to reason about what another agent will do indirectly by appealing to knowledge about what the other agent believes and wants--together with the principle that rational agents act in a way they believe will achieve their goals. We have been exploring various ways the notion of rational action has been formalized in the literature with a view towards finding a suitable version to incorporate into the AI planning paradigm.

During the last few months we have been looking at several potential applications domains, including robotics and automated routine business negotiation.

References

[1] Appelt, D.E., A Planner for Reasoning about Knowledge and Action, *Proceedings of the First Annual National Conference on Artificial Intelligence*, August 1980.

[2] Konolige, K., A First-Order Formalization of Knowledge and Belief for a Multiagent Planning System. SRI Technical Note 232, December 1980.

[3] Konolige, K. and N.J. Nilsson, Planning in a Multiple Agent Environment, *Proceedings of the First Annual National Conference on Artificial Intelligence*, August 1980.

[4] Rosenschein, S.J., Plan Synthesis: A Logical Perspective. *Proceedings of the International Joint Conference on Artificial Intelligence, Vancouver*, B.C., August 1981.

Formalization of Knowledge, Belief and Action for a Multiple-Agent Planning System -- Deductive Issues

Kurt Konolige
SRI International
Menlo Park, Ca. 94025

After an initial period of trying to formulate multiple-agent plans by extending existing planning systems [1], it was decided that problems involving the formalization of knowledge and action needed to be solved before an adequate planning system could be built. For several reasons, we took a syntactic approach to this formalization; the results are in [2]. This syntactic approach describes an agent's knowledge of the world as a theory in some first-order language, called the object language (OL); a meta-language (ML) can then be used to characterize partial information about an agent's knowledge, and to describe the inferences that an agent might make.

Currently we have begun to address some of the deductive issues involved in making inferences in the ML/OL system. In particular, since all deductions in the OL are simulated by appropriate axioms describing these deductions in the ML, even simple deductions of the OL become complicated deductions in the ML. To compensate for the loss of efficiency, of course, whole classes of OL proofs can be simulated by a single ML proof; but in the problems we are interested in solving, the loss of efficiency may be of primary importance. So we have developed a method of bypassing the simulation of OL deductions in the ML by running a theorem prover in the OL when we want to find the answer to some ML expression involving OL sentences. The technique is an extension of Weyhrauch's semantic attachment; it is possible to use this extended form of attachment even when the OL sentence is described with ML variables and uninterpreted functions. On initial evaluation, extended semantic attachment seems to work quite well; a version of the Wise Man Puzzle was solved in about 6 seconds of CPU time in INTERLISP on a DEC 2060.

References

[1] Kurt Konolige and Nils Nilsson, "Multiple Agent Planning Systems," Proceedings AAAI, Stanford, CA. 1980.

[2] Kurt Konolige, "A First-Order Formalization of Knowledge and Action for a Multi-Agent Planning System," SRI Tech Note 232, 1980.

The Use of Possible World Semantics in a Multiple Agent Planning System

Doug Appelt
SRI International
Menlo Park, Ca. 94025

My primary motivation for work in Distributed Artificial Intelligence is the need to account for the processes by which agents plan linguistic actions. A planning system called KAMP (Knowledge and Modalities Planner) [2, 3] has been developed that is capable of producing plans in which one agent produces utterances that will influence the mental state (i.e. beliefs and wants) of another agent, and that forms plans involving the cooperation of several agents in achieving a common goal. The work reported here is related closely to that of Allen, Cohen and Perrault [1, 4], and is related in spirit to the work reported by Davis at this workshop. Like Allen et al., I am interested in relating DAI work to natural language understanding and generation, and like Davis, I am interested in forming plans among cooperating agents. I start from the assumption that agents have incomplete rather than complete knowledge about the world, and that the agents are mutually aware of the actions they perform.

Any system for multiple agent planning must be based on a sound representation and deduction system that is capable of reasoning about what agents know, believe and want. One way of approaching the representation problem was proposed by Moore [6]. Facts about knowledge and belief can be expressed in an intentional logic, the possible worlds semantics of which is axiomatized in a first order theory. An agent knows P if and only if P is true in every possible world compatible with his knowledge. The effects of actions can be represented easily in this formalism as well, where possible worlds can be related by actions which transform one world into another.

The desirable features of this approach include the ability to represent and reason efficiently about both knowledge and ignorance, and the ability to integrate conveniently into the same formalism the description of actions and their effect on knowledge. An example of the type of deduction this approach works well on is where one wants to conclude ~Know(A, P) from Know(A, P -> Q) and ~Know(A, Q). Konolige [5] advocated in his workshop presentation a representation and deduction system based on the semantic attachment of multiple first order theories to a first order meta-language. There are many unresolved problems in controlling the inferences in a deduction system employing that representation, and although work is still in progress, I have decided that at this time, the possible worlds semantics approach to reasoning about knowledge and action offers the most promising foundation upon which to build a multiple agent planning system.

In spite of the good points in its favor, problems arise when the possible worlds formalism is used for planning. Instead of an explicit set of assertions about what an agent believes, the agent's beliefs are characterized implicitly by the set of possible worlds compatible with his knowledge. Since the soundness of the foundation upon which a multiple agent planning system is built is extremely important, it was decided to retain the possible worlds formalism and attempt to discover a heuristic means of facilitating plan generation. Since the possible worlds formalism is more suitable for the verification of plans than the discovery of plans, KAMP employs two descriptions of each action. One description is the axiomatization of the action in terms of relations between possible worlds. The other description is a STRIPS-like action summary which describes a set of normally intended knowledge state and physical state effects. The action summaries are used to heuristically guide the search for a plan by proposing plans that are likely to succeed, and the plans can then be verified using the logical axiomatization. If the verification proof fails, the failed proof tree can provide clues

to the planner about what went wrong and how to fix the problem.

KAMP has been used as the basis of a language planning system that is capable of producing plans involving two agents cooperating on a task, and producing speech acts as part of the plan which involve the integration of multiple illocutionary acts into a single utterance.

References

[1] Allen, James and C. R. Perrault, Participating in Dialogues: Understanding Via Plan Deduction, *Proceedings of the Second National Conference of the Canadian Society for Computational Studies of Intelligence*, 1978.

[2] Appelt, Douglas E., A Planner for Reasoning about Knowledge and Action, *Proceedings of the First Annual National Conference on Artificial Intelligence*, 1980.

[3] Appelt, Douglas E., Problem Solving Applied to Language Generation, *Proceedings of the Eighteenth Annual Meeting of the Association for Computational Linguistics*, 1980.

[4] Cohen, Philip and C. R. Perrault, Elements of a Plan Based Theory of Speech Acts, Cognitive Science, vol. 3, pp. 177--212, 1979.

[5] Konolige, Kurt, and N. Nilsson, Planning in a Multiple Agent Environment, *Proceedings of the First Annual National Conference on Artificial Intelligence*, 1980.

[6] Moore, Robert C., Reasoning about Knowledge and Action, SRI International AI Center Technical Report No. 191, 1980.

Invited Talk:

Natural Language Pragmatics and DAI

C. Raymond Perrault
Dept. of Computer Science
University of Toronto - Ontario, Canada 1827

There are two obvious ways in which research in DAI and Natural Language Processing (NLP) make contact. The first way is to consider NLP as a DAI problem, as done for example in the Hearsay-II system. The second approach, and the one taken here, is to consider communication in DAI systems, and what it shares with human communication. Our talk surveyed two key problems in the pragmatics of natural language and suggested that they may be of interest to designers of DAI systems: the form of the discourse context and the recognition of intention.

Drawing on transcripts (collected by Tennant) of dialogues between the PLANES question-answering system (developed by D. Waltz and his colleagues) and some users, we showed several ways in which these users expect the system to react to their unstated intentions. Users also depend on the earlier utterances and on those of the system. We examined five dimensions along which the behavior of NLP systems can be compared, and related these to the problems uncovered in the transcripts. These dimensions are (a) versatility, or the range of functions a system is supposed to perform, (b) discrimination, or its ability to recognize from the user's actions which function(s) the user intends it to perform, (c) context-dependence, or its ability to rely on previous utterances by user and system to determine what the user's intentions are, (d) helpfulness, or its ability to do more than the user conveys, and (e) non-single-mindedness, or its ability to consider several sources of intentions in deciding what to do.

Any improvement in NLP systems will require a better understanding of the form of the discourse context and of the communication of intentions. We briefly discussed some work we have done on these problems with Phil Cohen and James Allen. Some examples were given to support the explanation of context as shared beliefs, and to demonstrate a range of options open for the definition of sharing. Following suggestions of philosophers of language such as Austin and Searle, speech acts defined for planning and plan recognition systems have been a useful way of dealing with some of the intentions associated with linguistic utterances. Several possible definitions of speech acts to assert and request were presented, and related to the versatility and discrimination of the agents which perform and recognize them. The more a system is able to protect its beliefs and intentions from modification by external agents, the more complex the necessary speech act definitions become.

List of Workshop Participants

Alphatech
John Delaney

Advanced Information & Decision Systems
Chee Chong
Robert Drazovich

ARPA
Ronald Ohlander

Carnegie-Mellon University
Rick Rashid
Mark Fox

Defence Research Establishment Atlantic
Reid Smith

Information Sciences Institute
Lee Erman

Massachusetts Institute of Technology
Randy Davis
Dan Brotsky
Carl Hewitt
Bill Kornfeld
Judy Zinnikas

Xerox PARC
Rich Fikes

RAND
David McArthur
Perry Thorndyke

Rochester
James Allen
Jerry Feldman

Stanford Research Institute
Doug Appelt
Stan Rosenschein
Kurt Konolige

Stanford University
Mike Genesereth

University of Massachusetts at Amherst
Dan Corkill
Vic Lesser

University of Toronto
Ray Perrault