AI Magazine Volume 6 Number 3 (1985) (© AAAI)

Report on the 1984 Distributed Artificial Intelligence Workshop

Reid G. Smith

Schlumberger-Doll Research, Old Quarry Road, Ridgefield, Connecticut 06877

Abstract

The fifth Distributed Artificial Intelligence Workshop was held at the Schlumberger-Doll Research Laboratory from October 14 to 17, 1984 It was attended by 20 participants from academic and industrial institutions. As in the past,[1] this workshop was designed as an informal meeting It included brief research reports from individual groups along with general discussion of questions of common interest. This report summarizes the general discussion and contains summaries of group presentations that have been contributed by individual speakers

Distributed artificial intelligence (DAI) is concerned with cooperative solution of problems by a decentralized and loosely coupled collection of knowledge sources (KSs), each embodied in a distinct processor node. The KSs cooperate in the sense that no one of them has sufficient information to solve the entire problem; mutual sharing of information is necessary to allow the group as a whole to produce an answer. By decentralized we mean that both control and data are logically and often geographically distributed; there is neither global control nor global data storage. Loosely coupled means that the individual KSs spend most of their time in computation rather than communication. The processor nodes may be physically as well as logically distinct.

There are a number of reasons why one might choose to study such systems. Some researchers are motivated by the promise of building more powerful problem solvers, with greater speed, reliability (including tolerance of uncertain data and knowledge), and extensibility (the ability to scale the resources applied to a problem). Others are motivated by problem characteristics, such as natural spatial or functional distribution. Finally, some researchers are motivated by the observation that some issues in problem solving and reasoning (*e.g.*, communication and coherent cooperation) are brought into sharp focus in the distributed setting.

The main body of this report contains summaries written by individual attendees of the workshop. This is, in turn, followed by a brief summary of the general discussion.

[1]Three previous workshops have been summarized elsewhere (Davis, 1980, 1982; Fehling and Erman, 1983)

Project Reports

What Can AI Learn From Looking at Human Organizations?

We are primarily involved in two DAI projects. Each illustrates different aspects of how AI can benefit from looking at human organizations:

The first is a marketlike task scheduler for distributed computing environments. This system, called Enterprise (Malone *et al.*, 1983), allows otherwise unused personal computers on a network to perform tasks for other users. Like Smith and Davis's Contract Net protocol (Smith, 1981; Davis and Smith, 1983), this system is based on the metaphor of a market: Processors send out "requests for bids" on tasks to be done and other processors respond with bids giving estimated completion times that reflect machine speed and currently loaded files. Then tasks are assigned (possibly in parallel) to the best machines available at run-time (either remote or local). A prototype version of this system has been implemented in Interlisp-D to run on Xerox Dolphins, Dorados, and Dandelions connected by an Ethernet.

In a series of simulations of different load conditions and network configurations, the scheduling protocol used in Enterprise was found to be substantially superior to both random assignment of tasks and a more complex alternative that maintained detailed schedules of estimated start and finish times.

Project personnel: Thomas Malone, Richard Fikes, Michael Howard, Kenneth Grant. (Most of this project was carried out when Malone, Fikes, and Howard were at Xerox PARC. Fikes is now at IntelliCorp and the other project members are at MIT.)

The second project is a taxonomy of organizational structures and criteria for choosing among them This project has developed a taxonomy of different kinds of structures that can be used to organize parallel processing computer systems (Malone and Smith, 1984). The structures that are categorized include various kinds of hierarchies and various kinds of market-like organizations For example, the Enterprise system described above is an instance of a decentralized market. An alternative implementation of the system could have used a single centralized scheduling node on the network to keep track of the status of all processors and assign tasks to them as appropriate.

This project has also derived general mathematical results comparing the structures on criteria such as efficiency (*e.g.,* processing delay times) and flexibility (*e.g.,* reliability). For example, using rough estimates of parameters such as message transmission time and machine reliability in the environment in which the original Enterprise system was implemented, it appears that the decentralized market approach is superior to the alternative centralized approach.

Project personnel: Thomas Malone (MIT), Stephen Smith (University of Santa Clara).

—Thomas W. Malone

Cooperative Intelligent Systems

This research project focuses on the development of strategies for cooperative problem solving in complex, spatially distributed systems. Our objective is to produce guidelines for use of such strategies in specific operational applications. In particular, we are focussing on a system of remotely piloted vehicles (RPVs) on reconaissance or attack missions.

Each RPV has a set of functions that include communication over a noisy network, data gathering and fusion, formation/pattern flight in two dimensions, and defense suppression. A cooperative strategy is a set of rules for RPVs for combining their functions and achieving the group objective.

We have developed methods for RPVs to efficiently perform their functions, as well as studied strategies for RPV cooperation. So far we have experimented mainly with leader-based strategies. We are now working with more sophisticated strategies, particularly those involving independent decision making by individual RPVs Simulations of strategies are written in ROSS, an object-oriented language developed at Rand.

Our future research objectives include better understanding of the nature of plans, time, and continuity, better handling of incomplete or inconsistent information, and more effective techniques for simulating dynamic processes.

—Randy Steeb, Sanjai Narain, Stephanie Cammarata, and William Giarla

Coherence in Distributed Problem Solving[2]

Globally coherent behavior is the Holy Grail of distributed problem-solving network research. Obtaining coherent network activity without sacrificing node autonomy and network flexibility places severe demands on the local control component of each node Our approach to network coherence has been to provide each node with a high-level strategic plan that defines the node's network responsibilities. This strategic plan, represented as a network organizational structure, specifies in a general way the communication and control relationships among the nodes. The organizational structure increases the likelihood that nodes will be coherent in their behavior by limiting the range of options available to a node. However, the organizational structure must not limit the options of nodes too tightly, or the network's ability to quickly adapt to changing problem-solving situations and network characteristics will be lost. Sophisticated local node control plays a key role in this approach because of the need to further refine the options specified by the organizational structure based on short-term information about the current situation.

We are using the distributed vehicle monitoring testbed to explore our ideas (Lesser and Corkill, 1983). The testbed simulates a network of problem-solving nodes attempting to identify, locate and track patterns of vehicles moving through a two-dimensional space using signal detected by acoustic sensors. Each problem-solving node is an architecturally complete Hearsay-II system with knowledge sources and levels of abstraction appropriate for this task. The basic Hearsay-II architecture has been extended to include more sophisticated local control, knowledge sources for communicating hypotheses and goals among nodes, and data structures that specify the organizational role of a node (Corkill and Lesser, 1983).

[2]This research is sponsored, in part, by the National Science Foundation under Grant MCS-8306327 and by the Defense Advanced Research Projects Agency (DOD), monitored by the Office of Naval Research under Contract NR049-041

We are presently using the testbed to explore, through implementation and empirical studies, a variety of issues in the use of sophisticated control for network coordination. These include:

- **Knowledge-Based Fault-Diagnosis:** How can we detect and locate inappropriate system behavior. Inappropriate behavior includes problems caused by hardware errors, as well as inappropriate settings of the problem-solving parameters that specify strategic and tactical network coordination.
- **Distributed Load Balancing and Communication Strategies:** What specific data, processing results, and processing goals should be transmitted among the nodes? Given a high-level strategic plan for the allocation of activities and control responsibilities among nodes, there is still a need for the nodes to negotiate more localized, tactical decisions.
- **Organizational Self-Design:** What is an appropriate initial organizational structure for a particular problem-solving situation? We are working on the construction of an organizational structuring module that will select an appropriate structure based on situational parameters. This module will also modify the organizational structure to reduce the effect of hardware errors or an inappropriate organizational structure that may arise during the course of problem solving.

The goal of this research is to develop concrete information about the nature of coordination required for effective distributed problem-solving system performance where there are large numbers of nodes operating in a dynamically changing environment.

—Victor R. Lesser and Daniel D. Corkill

The MINDS Project: Multiple Intelligent Node Document Servers

Documents are used in computerized offices to store a variety of information, such as data, software, letters, graphics, and spreadsheets. This information is difficult to use, especially in large offices with distributed workstations, because document names are not unique and do not convey enough information about document content. In addition, users do not have perfect knowledge of either the documents in the system or the organization for their storage. The goal of the MINDS Project (Bonnell *et al.*, 1984; Huhns *et al.*, 1983; Yang *et al.*, 1985) is to develop a distributed system of intelligent servers that are used to organize and retrieve documents. Each server, implemented at a single-user workstation, is a knowledge-based system (KBS) that is an expert in the domain of "documents." The KBS has knowledge in the form of local documents and their abstractions, as well as metaknowledge about the locations of nonlocal documents. Based on principles of distributed artificial intelligence, these KBSs share knowledge, metaknowledge, and tasks to cooperate in processing queries from users.

A user accesses documents by issuing a query to his local KBS, which initiates a search. A central document directory containing all the relevant information regarding each document and its location could be used to process system-wide queries. However, a large network would result in a voluminous central directory, a vulnerable system, and prohibitively large search costs. Also, when content-based queries are processed, too many documents may satisfy the criteria for the search.

A distributed system with the dynamic, task-driven, partial-view-integration scheme of the MINDS system produces efficient searches and relevant responses. In this system, each KBS has its own metaknowledge to help guide searches, which proceed according to a best-first strategy. The metaknowledge is specialized for the requirements of the workstation where it is located. Through the experiences of interacting with other KBSs and its local user, each KBS learns and reasons about the knowledge contained at other workstations and modifies its metaknowledge accordingly. As a result, the same content-based query would not necessarily generate the same response if issued at different workstations. Each response would consist of the documents most relevant to the user at that workstation, ordered from highest to lowest relevance.

A unique method for modeling the organization and distribution of metaknowledge using an entity-relationship diagram has also been developed. This diagram forms the basis for the system architecture, which allows queries to be processed in parallel by multiple workstations. A query is first decomposed at its originating workstation, and then query-processing subtasks are distributed among other workstations in the network according to the current metaknowledge. When a workstation receives a subquery, it sends a response to the originator of the subquery for result synthesis. If the subquery is content-based, the workstation also sends its relevant metaknowledge to the originator of the subquery, resulting in an updating of the originator's metaknowledge.

Another aspect of MINDS is the use of confidence factors to control the system-wide processing of queries. The KBS at each workstation provides a confidence factor for each association between two users and a keyword. This confidence factor provides an ordering of the search for documents. It reflects: How broad the information at a workstation pertaining to a specific keyword; how useful documents concerning this keyword have proven to be in the past; and how recently the workstation has acquired its information.

MINDS also incorporates a scheme for learning based on the creation and deletion of pointers among workstations, keywords, and documents as well as on the mainte-

nance of attribute values, including the above confidence factor. This metaknowledge is initialized with default values and changed as the workstation learns about its neighbors through interaction. Thus, the system is self-initializing.

—Michael Huhns, Ronald Bonnell, Larry Stephens, and Uttam Mukhopadhyay

Distributed Artificial Intelligence at GTE Laboratories

The issues of interest to GTE Laboratories are communication between and cooperative planning/action among independent intelligent agents connected on a communication network. To gain familiarity with the issues and their associated problems, the research plan calls for several experiments to be designed exemplifying situations where such capabilities would be useful. In addition to the experimental effort, GTE Laboratories will be exploring models of intelligent agents to develop a framework for communication and cooperation in the general case. As these efforts develop, their results will be merged, hopefully creating a workable strategy for constructing cooperative, autonomous systems.

An experimental testbed is being constructed that will permit a wide variety of issues to be explored. Among the experiments being considered is one which will explore issues related to the task of real-time cooperative planning and action. By varying the content, cost, and availability of communication, the experiment is intended to measure the effectiveness of the team under a variety of conditions. From this, strategies for when and how to communicate (and eventually, how to form and change organizations to meet the task conditions) may be developed. In addition to the testbed, GTE Laboratories is investigating issues related to how these techniques can be applied to problems in development, control, and maintenance of communication networks.

Proposed theories of agent and organization will be realized as computer models are developed and their performance assessed. Although these models are initially driven by performance models of humans and human coalitions, imitation of human activity is not the evaluation criterion; realization of intelligent interagent cooperation is. This work will require analysis of computer models themselves: in particular, constraint expression and propagation will be examined as a general descriptive mechanism for command and control processes. Moreover, how to represent the interrelated purposes of an agent and the relationship of these purposes to plans and actions will be studied. A key question is one of the evolution of structure: when and how do a set of purposes and a present situation combine to direct the creation of a new cooperative link between independent agents?

—W. Frawley and R. Worrest

A Formal Framework for Rational Interaction

The development of intelligent agents presents opportunities to exploit intelligent cooperation. Before this occurs, however, there is a need to understand the mechanisms of interaction. We are developing at Stanford a formal analysis of rational interaction; that is, interaction that would occur among rational agents. Consideration is being given to single-event and recurring interactions, interactions with and without communication, and interactions in which the agents have identical or disparate goals.

The problem and the benevolent agent assumption. Research in AI has focused for many years on the problem of a single intelligent agent. The presence of multiple agents, however, is an unavoidable condition of the real world, and when people plan their own actions, they have to take into account the potential (harmful or helpful) actions of others. When intelligent agents venture into real world domains, they will need to interact flexibly with one another and with people.

To understand how agents ought to interact with one another, we need a formal framework in which to model general interaction (*i.e.,* interaction among agents with potentially conflicting goals). This is a task that remains largely untouched by the AI community, where researchers have uniformly adopted what we call the "benevolent agent" assumption: All agents are assumed to have identical or nonconflicting goals and to freely help each other (with limited exceptions). We are, instead, examining the "conflicting agent" metaphor: We have developed a framework that allows agents to have conflicting goals and allows us to study when these self-interested agents actually will cooperate and compromise.

Agents with identical goals, communication strategies. Early work focused on the case of communicating agents that have identical goals, but potentially distinct (and conflicting) information about the world (Rosenschein and Genesereth, 1984a). We have isolated strategies of information transfer that will cause the agents to converge to identical plans, and identified strategies that will not cause convergence. We have also examined what role "lying" (*i.e.,* the transfer of locally inconsistent information) might play in such interactions, and its limited utility.

Agents with distinct goals, no communication allowed. Later work (Genesereth *et al.,* 1984a, 1984b) focused on the case where agents with potentially conflicting goals had to interact, and no communication was possible (*i.e.,* after initial joint recognition of the interaction). A hierarchy of rationality assumptions was developed, and various types of behavior were shown to provably follow from each level of rationality. The notion of "common rationality"

(where agents use identical decision procedures in choosing their actions) was introduced, and certain types of positive behavior were shown to result from this assumption.

Agents with distinct goals, promises allowed. Our latest work (Rosenschein and Genesereth, 1984b) has introduced into the "conflicting agents" model the notion of binding promises and deal making. Agents with distinct goals are assumed to be capable of promising particular actions, contingent on certain conditions being met. Our research has answered the question of what kinds of promises are rational, given assumptions (once again) about the rationality of other agents. It has also shown the power of communication: using a deal-making mechanism, agents are able to coordinate and cooperate more easily than in the communication-free model. In fact, there are certain types of interactions where communication allows mutually beneficial activity that is otherwise impossible to coordinate.

Conclusion. DAI research has so far operated under a self-imposed set of blinders. Work has progressed on narrow issues of agent cooperation, but the assumption of "agent benevolence" has always been present. DAI has by and large ignored the issue of why agents might cooperate with one another, and how they would cooperate if truly conflicting goals were present.

"Agent benevolence" is a special case of general interactions, and, unfortunately, it does not appear that many solutions to this special case will apply to the more general case. Progress made studying the general case, on the other hand, should translate into solutions for the restricted case. While work certainly should continue on the special case, researchers must recognize that progress will be useful only in domains where all the agents have an identity of interests, where they are, in some sense, a single problem-solving entity that has become fragmented. To deal with more general domains, such as autonomous land vehicles in a military setting, or an automated personal secretary looking after the scheduling needs of its boss, a more general paradigm of interaction must be pursued. We have to allow for true conflict, negotiation and compromise among intelligent agents.

—Michael R. Genesereth, Matthew L. Ginsberg, and Jeffrey S. Rosenschein

Knowledge Representation and Inference in a Parallel Evidential Framework

Work at Rochester on distributed artificial intelligence covers a broad spectrum of research areas including low, intermediate, and high-level vision, motor control, natural language understanding, knowledge representation, and inference. The emphasis is on developing solutions in terms of connectionist networks (*i.e.,* massively parallel networks comprising highly interconnected, but extremely simple processing elements). The work discussed at the workshop (in collaboration with Prof. J. A. Feldman) concerns knowledge representation and inference (Feldman and Shastri, 1984; Shastri and Feldman, 1984; Shastri, 1985).

The computational cost of gathering, processing, and storing information about a complex and constantly changing environment makes it impossible to maintain complete knowledge. However, the need to act on available information compels an agent to make decisions (inferences) based on incomplete knowledge. This underlines the necessity of formalizing inference structures that can deal with incompleteness and uncertainty. Furthermore, an agent must often act in "real time" (*i.e.,* within a fixed amount of time). This points out the necessity of developing representation and inference formalisms that have efficient implementations.

We are developing an evidential approach to knowledge representation and inference wherein the principle of maximum entropy is applied to deal with uncertainty and incompleteness. An evidential approach permits the association of numeric quantitites with assertions to indicate their degree of belief, and has long been used in expert system design. Our work attempts to demonstrate that it is possible to adopt an evidential approach in solving some well-known problems in knowledge representation. The work focuses on a representation language that may be viewed as an evidential extension of inheritance hierarchies and develops a formal theory of evidential inheritance and recognition within this language. The theory applies to a limited, but we think interesting, class of inheritance and recognition problems, including those that involve exceptions and multiple hierarchies. The resulting theory may be implemented as connectionist networks that can solve inheritance and recognition problems in time proportional to the depth of the conceptual hierarchy.

The proposed evidence combination rule is incremental, commutative, and associative and hence shares most of the attractive features of the Dempster-Shafer evidence combination rule. Furthermore, it is demonstrably better than the Dempster-Shafer rule in the context of the problems we are addressing.

—Lokendra Shastri

The Challenge of Open Systems

Large-scale distributed systems are evolving rapidly because of the following developments:

- The burgeoning growth of the number of personal computers.
- The development of local and national electronic networks.

- The widespread requirement of arm's-length transactions among agencies and organizations.

Systems of interconnected and interdependent computers are qualitatively different from the relatively isolated computers of the past. This article discusses some of the implications and constraints imposed by these new developments. Such open systems uncover important limitations in current approaches to artificial intelligence, such as problem spaces and logic programming. They require a new approach more like organizational design and management than current approaches.

In practice human knowledge of a physical system cannot be consistently described in axioms. Every physical system is open in the sense that it is embedded in a larger physical environment that interacts asynchronously. The Japanese economy, the US Supreme Court, and the human kidney are good examples of open systems. Open systems are not totally in control of their fate. By contrast closed systems (like Peano arithmetic and point set topology)can be exactly characterized by rules and laws.

Proponents of logic programming have maintained that it provides a suitable basis for all programming and is, in fact, the programming paradigm for the future. Logic programming has some fundamental limitations that preclude its becoming a satisfactory programming methodology, because it does not deal adequately with empirical knowledge. It is inadequate for the needs of open systems because it is based on logical operations instead of communication primitives and because it is based on logical reasoning instead of due-process reasoning. Decisions in open systems are made by processes whose outcomes cannot be predicted in advance. They are justified by agreements to act in certain ways. Justification by agreement stands in contrast to justification by logical proof. Decision making in the physical world involves dealing with conflicting and contradictory information in a way that does not fall within the scope of making decisions by logical proof. By their inherent structure proofs do not take contrary arguments into account. PROLOG suffers from the additional limitation of being based on "negation as failure," which limits it to a closed-world assumption that is incompatible with the nature of open systems.

We need foundations for intelligent systems based on principles of commutativity, pluralism, accessibility, reflection in practice, and due-process reasoning. Logical reasoning is a useful module in the repertoire of an intelligent system, but it is not the whole show. Application of the principles used in designing and managing large-scale organizations will be fundamental to the future of open systems.

—Carl Hewitt

Applications of Economic Organization and Interaction to Resource Sharing and Optimization in Computer Networks

Our current work is aimed at the development and experimental investigation of two decentralized approaches towards engineering algorithms for optimizing resource allocation and access in distributed systems. The term "engineering" is emphasized because we believe these approaches can be used to systematically develop optimization algorithms for a large class of resource allocation and access problems.

Our approaches are based upon the belief that many of the decentralized models and methods previously developed to study another distributed environment, the perfectly competitive economic marketplace, have important applications within the field of distributed systems. In the past three decades, mathematical economists have developed elegant normative models describing how resources may be optimally distributed by humans in an economy in an informationally and computationally decentralized manner. In our work, we view the computational agents in the distributed environment as an artificial society of processors and use these models of resource sharing and optimization in human societies as blueprints for engineering similar mechanisms in distributed systems.

In order to study, develop, and test these ideas within a specific context, we are currently implementing these algorithms in a simulated network environment and experimenting with their use for solving the problems of optimizing multiaccess channel protocols and optimally allocating files among the nodes in a distributed system.

Jim Kurose

Overview of the FAIM-1 Multiprocessing System

Some initial observations. Much of the presently available AI technology cannot be realistically applied in the solution of real problems because of the inherent limitations of its performance. This results partly from the sequential nature of the programs, and partly from to the limitations of the machines which run these programs. Significant performance increases will be necessary to make AI solutions viable for large scale machine intelligence applications. The normal technology mechanisms of faster circuits and packaging methods applied in the context of conventional systems will not be sufficient to achieve more than a factor of 100 performance increase. The only viable alternative is to develop real systems that exploit very

high levels of concurrency. The performance of such systems is limited by human imagination rather than by hard technological barriers.

The transition to high-utility concurrent systems will not be easy. Concurrent systems inherently contain more organizational overhead than traditional sequential systems. The most important factor of this overhead is communication. An ensemble of concurrent tasks must cooperate in the overall solution and such cooperation often requires high levels of communication, which must be supported in such a way that the resultant overhead does not negate the benefits of the parallelism. The other barrier to this transition is that the sequential problem solving methods that AI practitioners have developed over the past 20 years may be invalid in a concurrent environment. The development of a new and highly concurrent problem solving methodology will be difficult. If high quality machine systems are not available to run the applications at their intended performance levels, then the applications programmers will not be sufficiently motivated to develop these methods.

The FAIM-1 multiprocessing system (Davis and Robison, 1985) is an attempt to provide both an ultra-performance machine built from advanced technology components and packaging and a set of programming tools to enhance the programming process. In general, the FAIM-1 system provides a radical architecture capable of exploiting very high levels of concurrent operation, while requiring only a minor level of change at the programming language level to incorporate concurrency.

Components of the FAIM-1 System. The FAIM-1 system provides:

- A computational model that is suitable for highly concurrent symbolic problem solving. The model is based on objects which communicate by passing messages.
- A Programming language OIL that implements the model and allows the concurrency inherent in AI problems to be readily expressed and subsequently exploited as a run-time performance mechanism.
- A Programming environment that provides a set of programming tools similar in quality to those currently used for high-quality sequential program development. Programmers will not make the conceptual shift to concurrent problem solving if the tools do not meet a minimum level of acceptability.
- A processing element that efficiently supports the language primitives and the communication-based concurrency mechanism of the computational model. Each element behaves as an independent processing site and is a medium-grain processor architecture that has been developed specifically for symbolic computation. The transistor budget has been reallocated in a way that is appropriate for AI programming and is significantly different from the traditional partition that has been in vogue for numeric computation.
- A communication topology that interconnects an ensemble of processing elements in a scalable manner. This topology is consistent with modern high-speed packaging technology and permits an arbitrary number of processing elements to be connected into a concurrently cooperating ensemble.
- A resource allocation mechanism that maps useful concurrency extant in the program onto the available physical resources in a way that this concurrency can be exploited at run-time as a performance mechanism. This mechanism permits the programmer to develop a concurrent solution in a manner that is imposed by the nature of the problem and not by the structure of the machine. We should not have to revert to the dark ages of machine-dependent programming to incorporate concurrency.
- A run-time system supporting the highly-concurrent-evaluation style of the machine.

Status. The OIL programming language has been defined. Object behaviors can be either concurrent logic programs or concurrent Lisp programs. An interpreter for the OIL language is being developed on a Symbolics 3600 LISP machine, which will also serve as the host machine for the FAIM-1 prototype. Development of the OIL compiler and programming environment is still in the initial phase, and serious implementation will not begin until late in 1985.

The architecture of FAIM-1 is a multiply twisted hexagonal mesh. The architecture of the processing element (Hectagon), has been specified, and implementation is proceeding on the six subsystems:

- ISM: Instruction Stream Memory (Coates, 1985).
- CxAM: Context Addressable Memory (Brunvand, 1984; Robinson, 1985).
- SRAM: Scratchpad RAM.
- FRISC Fanatically Reduced Instruction Set Computer (the processor).
- SPUN: Streamed Pipelined Unifier (hardware unification support).
- POST-OFFICE: communication support.

Some of these subsystems are being implemented as custom CMOS VLSI components, while others are being developed as board-level prototypes using high-speed conventional components. Eventually all of the subsystems will be built from custom integrated circuits. The eventual packaging will be immersion-cooled, wafer-scale hybrid CMOS.

The resource allocation mechanism is based on a graph-embedding style where the parameterized program graph is massaged to fit the parameterized machine graph

The code is a combination of graph transformation heuristics and simulated annealing.

We hope to have the initial prototype available for evaluation in early 1987. A full-system simulation is targeted for early 1986. Prospective programmers will be able to use the simulator to evaluate both the design of the FAIM-1 system and the applicability of their concurrent solutions.

—A. L. Davis

A Distributed Vision Experiment

To highlight the central problems in cooperation among (distributed) artificially intelligent agents, we conducted a "distributed vision" experiment with intelligent, non-artificial agents: each workshop participant was given one piece of a Dali print, and the group was asked to derive cooperatively a description of the complete picture.

Since the time was too short to converge on a single description (if such convergence was indeed possible), the experiment yielded a wide variety of descriptions at "nodes" throughout our network. The issues that arose (*i.e.,* the complaints that were voiced) include the following:

- The need for common language/protocols/conventions for communication (*e.g.,* precisely what color is "brown", how long is "long?").
- The importance of assessing the potential utility to others of various possible descriptions, levels of detail, and so on, to avoid frustrating them with useless information.
- The significance of effectively using the evidence provided by others (*i.e., recognizing important clues to* the picture's contents, and disregarding irrelevant information).

The last of these three is of prime interest to us; it appears that none of the standard mechanisms for belief revision (*e.g.,* Bayes' rule, Shafer's rule of combination) is adequate for an intelligent system, as none permits consideration of the utility (*i.e.,* informative value, predictive value) of a potential change in beliefs. Our current research is aimed at developing a decision-theoretic model of belief revision in which deliberation weighing the utility of candidate beliefs against the possibility of their falsehood is the source of changes to belief systems.

—Monnett Hanvey

Meta-Encoding of Information in an Object-Oriented Distributed Problem Solver

At the workshop, we described a distributed problem-solving architecture, based on the Contract Net protocol (Smith, 1981; Davis and Smith, 1983), implemented in Strobe (a structured-object programming system) (Smith, 1983; Schoen and Smith, 1983; Smith, 1984) D-Machines and Vaxes, connected via Ethernet. The architecture has been used to experiment with well-log zoning. (In this context, a zone is an interval of the borehole data that has been concluded to be coherent according to some criterion *e.g.,* lithology).

Within our architecture, the grain of internode communication is the object. Nodes selectively transmit object descriptions that can be inserted by other nodes into their local knowledge bases. Meta-encoding is used to select and abstract the objects to be transmitted. By meta-encoding we mean encoding information about data in a way that is less domain-specific than the data itself. For instance, we encode each task to be executed by a processor as an object (*e.g.,* ZoningTask) that includes a number of slots (*e.g.,* Log, Zones, ZoningCriterion, ZoningMechanism, and ZoningParameters). Each slot is further annotated (using Strobe facets) with information (meta-data) that indicates the way in which it is related to the task (*e.g.,* the Log slot is annotated with a Role facet whose value, Input, is interpreted to mean that the slot contains data necessary to execute the task). This annotation allows the system to identify the information necessary to execute any task, without additional task-specific information (like slot names). The utility of meta-encoding is that it permits the use of a relatively small vocabulary of domain-independent annotations to cover an arbitrarily large set of domain-specific attributes.

To date, we have found it useful to be able to isolate information about tasks according to the following criteria:

- Input: The data required to execute the task (*e.g.,* Log, a particular type of log data).
- Output: The data produced by executing the task (*e.g.,* Zones, the detected zones).
- Transferability: Information necessary for executing the task either can or cannot be transmitted between processor nodes (*e.g.,* code vs hardware devices).
- Information: Attribute the specific aspects of input or output data that are to be communicated to other nodes (*e.g.,* the depth interval of a detected zone).
- Cooperation: Attribute the parameters that control dissemination of results (*e.g.,* when to advertise conclusions).

The object-oriented framework also helps to provide simple examples of how different views of the same result can come to be held by different nodes. For example, if NodeA communicates a single result object to NodeB, then NodeB's view of that object may differ from that held by NodeA because message handlers that would be inherited and extend the behavior of the object in NodeA may not be present in NodeB.

—Eric Schoen and Reid Smith

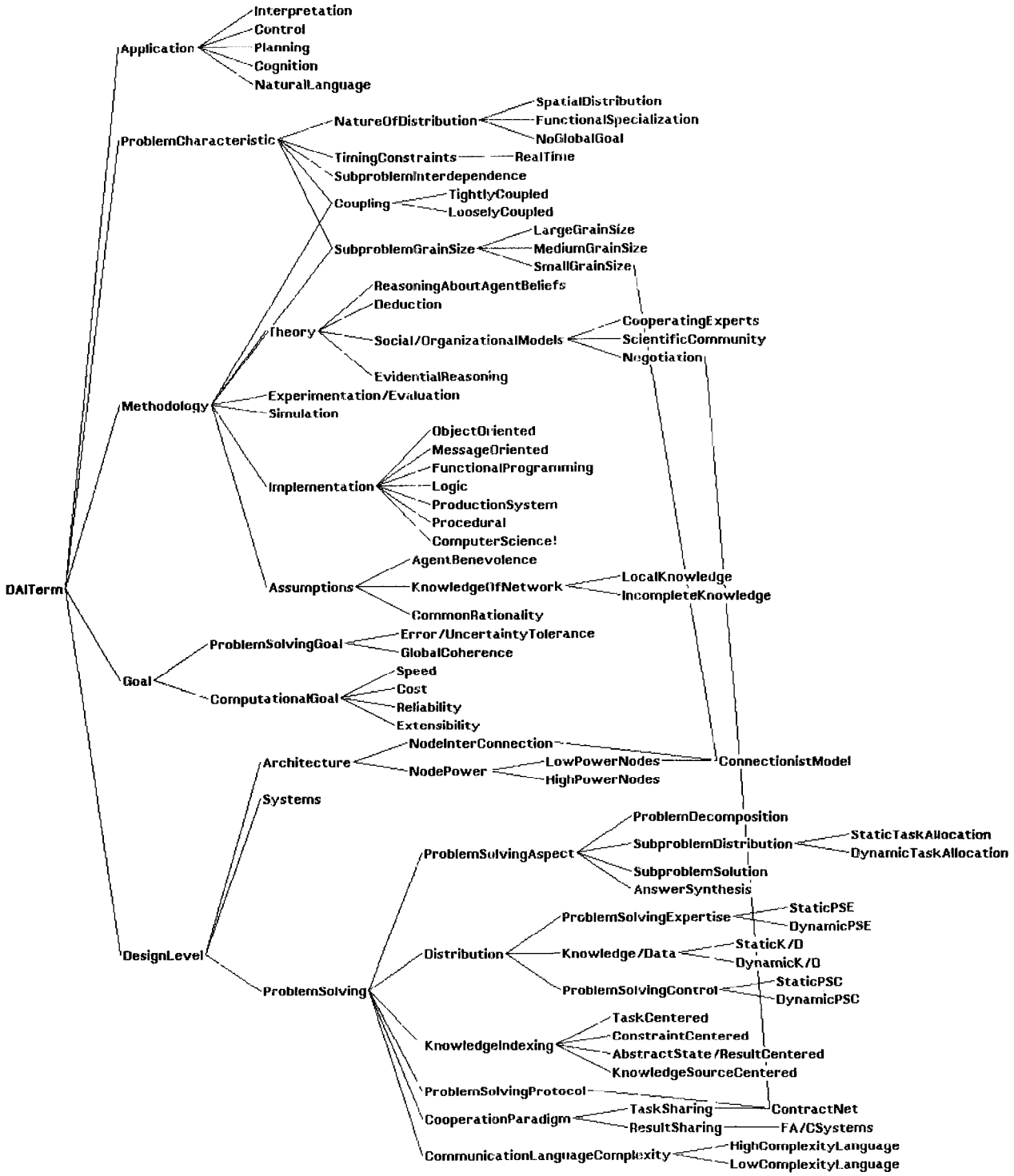
DAITerm
Application
Interpretation
Control
Planning
Cognition
NaturalLanguage
ProblemCharacteristic
NatureOfDistribution
SpatialDistribution
FunctionalSpecialization
NoGlobalGoal
TimingConstraints
RealTime
SubproblemInterdependence
Coupling
TightlyCoupled
LooselyCoupled
SubproblemGrainSize
LargeGrainSize
MediumGrainSize
SmallGrainSize
Methodology
Theory
ReasoningAboutAgentBeliefs
Deduction
Social/OrganizationalModels
CooperatingExperts
ScientificCommunity
Negotiation
EvidentialReasoning
Experimentation/Evaluation
Simulation
Implementation
ObjectOriented
MessageOriented
FunctionalProgramming
Logic
ProductionSystem
Procedural
ComputerScience!
Assumptions
AgentBenevolence
KnowledgeOfNetwork
LocalKnowledge
IncompleteKnowledge
CommonRationality
Goal
ProblemSolvingGoal
Error/UncertaintyTolerance
GlobalCoherence
ComputationalGoal
Speed
Cost
Reliability
Extensibility
DesignLevel
Architecture
NodeInterConnection
NodePower
LowPowerNodes
HighPowerNodes
ConnectionistModel
Systems
ProblemSolving
ProblemSolvingAspect
ProblemDecomposition
SubproblemDistribution
StaticTaskAllocation
DynamicTaskAllocation
SubproblemSolution
AnswerSynthesis
Distribution
ProblemSolvingExpertise
StaticPSE
DynamicPSE
Knowledge/Data
StaticK/D
DynamicK/D
ProblemSolvingControl
StaticPSC
DynamicPSC
KnowledgeIndexing
TaskCentered
ConstraintCentered
AbstractState/ResultCentered
KnowledgeSourceCentered
ProblemSolvingProtocol
ContractNet
CooperationParadigm
TaskSharing
ResultSharing
FA/CSystems
CommunicationLanguageComplexity
HighComplexityLanguage
LowComplexityLanguage

Figure 1

Summary

A diversity of approaches to DAI has been apparent at each workshop. As a partial response, during the group discussion at the fifth workshop, we attempted to lay out a common language for comparing and contrasting DAI systems and results. We also hoped that at least the exercise, if not the language itself, would help to clarify matches between problems and approaches and suggest fruitful experiments. (An underlying theme was: What can one take away from the existing research to apply to a new problem?)

The result of the discussion is the primitive and incomplete graph shown in shown in Figure 1 It shows the classes of applications that have been considered to date, the characteristics of problems that have led to a desire for a distributed approach, the various methodologies that have been used, the goals that have been set out for systems, and the various levels at which research has been carried out. A link between two nodes in the graph simply means that they are related. The graph is included here more as a stimulus for future discussion than as a completed thought.

Workshop Participants

Al Davis (SPAR), *Schlumberger*
Dan Corkill, *U. MA at Amherst*
Lee Erman, *Teknowledge*
Bud Frawley, *GTE*
Mike Genesereth, *Stanford*
Monnett Hanvey, *Columbia*
Carl Hewitt (AI Lab), *MIT*
Mike Huhns, *U. of SC*
Jim Kurose, *U. MA at Amherst*
Victor Lesser, *U. MA at Amherst*
Tom Malone (Sloan School), *MIT*
Sanjai Narain, *Rand*
Hamid Nawib (Lincoln Laboratory), *MIT*
Jeff Rosenschein, *Columbia*
Arvind Sathi, *Columbia*
Eric Schoen, *Stanford*
Lokendra Shastri, *Rochester*
Reid Smith (SDR), *Schlumberger*
N. Sridharan, *BBN*
Ralph Worrest, *GTE*

References

Bonnell, R D , Huhns, M N , Stephens, L M , & U Mukhopadhyay. (1984) MINDS: multiple intelligent node document servers Proceedings of the IEEE International Conference on Office Automation, 125-136

Brunvand, E. L (1984) Context Addressable Memory for Symbolic Processing Systems. Masters Thesis, University of Utah.

Coates, W (1985) The Instruction Stream Memory. Schlumberger Palo Alto Research Artificial Intelligence Laboratory.

Corkill, Daniel D , & Lesser, Victor R. (1983) *The use of meta-level control for coordination in a distributed problem solving network* IJCAI-8, 748-756

Davis, R (1980) Report On The Workshop On Distibuted AI. *SIGART Newsletter*, 73:42-52

Davis, R (1982) Report On The Second Workshop On Distibuted AI SIGART Newsletter, 80: 13-23

Davis, A L & Robison, S V. (1985) *The FAIM-1 multiprocessing system* IEEE COMPCON, pp 370-375

Davis, R., & Smith, R G. (1983) Negotiation as a Metaphor for Distributed Problem Solving. *Artificial Intelligence* 20:63-109

Fehling, M & Erman, L. (1983) Report On The Third Annual Workshop On Distibuted Artificial Intelligence. *SIGART Newsletter*, 84:3-12

Feldman, J. A & Shastri, L (1984) Evidential inference in activation networks Proceedings of the Sixth Annual Cognitive Science Conference

Genesereth, M R., Ginsberg, M. L., & Rosenschein, J S (1984a) Cooperation without Communication HPP Report 84-36, Heuristic Programming Project

Genesereth, M R., Ginsberg, M L , & Rosenschein, J S. (1984b) Solving the Prisoner's Dilemma HPP Report 84-41, Heuristic Programming Project.

Huhns, M. N., Stephens, L M., & Bonnell, R D. (1983) *Control and cooperation in distributed expert systems* Proceedings of IEEE Southeastcon

Lesser, Victor R & Corkill, Daniel D (1983) The Distributed Vehicle Monitoring Testbed: A Tool for Investigating Distributed Problem Solving Networks *AI Magazine* 4(3)), 15-33

Malone, T W & Smith, S A (1984) Tradeoffs in designing organizations: Implications for new forms of human organizations and computer systems Working paper no CISR WP 112 (Sloan WP 1541-84), Center for Information Systems Research, MIT, March,

Malone, T W., Fikes, R E , & Howard, M. T (1983) Enterprise: A market-like task scheduler for distributed computing environments Working paper no CISR WP 111 (Sloan WP 1537-84), Center for Information Systems Research, MIT.

Robinson, I (1985) The Pattern Addressable Memory Schlumberger Palo Alto Research Artificial Intelligence Laboratory,

Rosenschein, J S & Genesereth, M R (1984a) Communication and Cooperation HPP Report 84-5, Heuristic Programming Project

Rosenschein, J. S & Genesereth, M R (1984) Deals Among Rational Agents. HPP Report 84-44, Heuristic Programming Project

Schoen, E & Smith, R G (1983) *Impulse, A Display-Oriented Editor for Strobe* AAAI-83, pp 356-358

Shastri, L (to appear) Evidential reasoning in semantic networks: a formal theory and its parallel implementation Computer Science Department, University of Rochester

Shastri, L & Feldman, J A (1984) Semantic Networks and Neural Nets Tech Rep TR-131, Computer Science Department, University of Rochester,

Smith, R G (1981) A framework for distributed problem solving. Ann Arbor, MI: UMI Research Press

Smith, R G. (1983) Strobe: Support For Structured Object Knowledge Representation IJCAI-8, pp 855-858.

Smith, R G (1984) Structured Object Programming In Strobe Research Note SYS-84-08, Schlumberger-Doll Research.

Yang, J. D , Huhns, M N., & Stephens, L. M (to appear) An Architecture for Control and Communications in Distributed Artificial Intelligence Systems IEEE Trans on Systems, Man, and Cybernetics.