Motoda, H., Mizoguchi, R., Boose, J. and Gaines, B. R. (1991). Knowledge acquisition for knowledge-based systems. IEEE Expert 6, 53-64.

Guest Editors' Introduction

Knowledge Acquisition for Knowledge-Based Systems

Hiroshi Motoda, Hitachi
Riichiro Mizoguchi, Osaka University
John Boose, Boeing Computer Services
Brian Gaines, University of Calgary

THE RESULTS OF THE FIRST JAPANESE KNOWLEDGE ACQUISITION FOR KNOWLEDGE-BASED SYSTEMS WORKSHOP PROVIDE BOTH AN OVERVIEW OF THE FIELD AND AN INTRODUCTION TO IEEE EXPERT'S *SPECIAL SERIES ON KNOWLEDGE ACQUISITION.*

ACQUIRING THE KNOWLEDGE to support an expert system is one of the key activities in knowledge engineering. Knowledge acquisition is not just a single idea; it is closely related to research in more general knowledge-system issues, and it covers a wide range of technologies. The work reported at the first Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop reflected this breadth.[1] Much of the work involved major advances in knowledge-systems research, such as task-specific architectures, classification problem solving, knowledge representation, knowledge compilation, and a variety of learning techniques both deductive and inductive. Our discussion here of the results presented at JKAW will provide an effective overview of the field and a solid introduction to *IEEE Expert*'s special series on knowledge acquisition.

Held in October 1990, JKAW was the tenth in a series of international knowledge-acquisition workshops and the first in the new Pacific Rim series. The workshop consisted of two open days of tutorials and invited talks at the Kyoto International Conference Hall and three closed days at the Hitachi Advanced Research Laboratory in Saitama.

Although the areas of work within knowledge acquisition are varied, some distinct categories were identifiable at the conference. Several talks in both the open and closed sessions focused on the relationship between knowledge, tasks, and domains. Researchers dealt both explicitly and implicitly with the idea of knowledge acquisition as domain modeling for different kinds of tasks. Several people also examined the problem of learning classification rules from examples. Other topics included using explanation-based learning for acquiring knowledge; using "knowledge acquisition support structures," such as hypertext and preprocessed natural-language text, as intermediate structures that can be accessed by a knowledge engineer for further formalization; and techniques and computer support for interviewing. There was also substantial interest in the knowledge acquisition front-end and the question of how to simplify the process of getting knowledge into the machine. Interviewing and the analysis of natural-language text are the major areas of research here.

The open session

The workshop started with an introduction by the Kansai Institute of Information Systems, followed by tutorials by Brian Gaines and John Boose and invited talks by B. Chandrasekaran, William Clancey, Thomas Gruber, Shigenobu Kobayashi, John McDermott, Hiroshi Motoda, Ross

0885/9000/91/0800-0053 $1.00 © 1991 IEEE

Quinlan, and Alain Rappaport. The session ended with closing remarks by Boose. More than three hundred people attended the open sessions.

Many talks centered on appropriate representations and reuse of various forms of knowledge to facilitate knowledge acquisition.

Foundations of knowledge acquisition. Gaines has developed a comprehensive model of the social and organizational aspects of knowledge acquisition, which he used to identify critical issues. He discussed the formation of expertise and the relationship among experts, clients, and systems, and he described a virtual-machine hierarchy and stressed the importance of portability and the development environment.

Gaines said that delivered systems integrate users and conventional technology in addition to knowledge-based system support tools. In his view, the knowledge acquisition bottleneck is still very much with us. We lack the variety of tools needed to help knowledge engineers transfer knowledge to shells, the shells are limited in the kinds of tasks they can help us with, and we have problems in validating, maintaining, and upgrading knowledge bases.

Nevertheless, there has been considerable progress. The first generation of knowledge acquisition tools was based on interviews by humans. The second generation included computer-based tools that help in the acquisition process. Future trends include more automation, integration, inclusion of multiple sources of knowledge, and support for verification, validation, and maintenance (the life cycle). In Gaines' view, third-generation tools will support a wide range of complementary tools and techniques. Development and delivery problems include diversity, system evaluation, access, user interfaces, standards, representations, and portability.

Tools, methods, and mediating representations. Boose described knowledge acquisition problems as well as the role of mediating representations — problem-modeling languages that help bridge the gap between the expert/knowledge engineer and computer implementations. Such representations should let an expert easily and naturally model the task and problem-solving process. In a sense, most expert-system and knowledge acquisition research in the last decade is in this area.

Boose compared manual and computer-based knowledge acquisition tools and techniques. Manual techniques include brainstorming, interviewing, and protocol analysis techniques. Semiautomated methods include shells: knowledge acquisition interfaces that use the structure of the generic problem to help structure and represent knowledge. Examples here include Heracles for heuristic classification, the Classification Structure Representation Language for hierarchical classification, and Aquinas for general analysis problems. In addition to these tools, which support knowledge acquisition for particular types of problems, there are tools that support it more generically. Such tools support interviewing, textual representation (such as hypertext) and modeling, acquisition of knowledge from multiple experts, and so on. Finally, there is a body of research and tools that support fully automated knowledge acquisition based on machine induction for concept learning (which is important for classification problems).

KNOWLEDGE ACQUISITION IS NOT JUST A SINGLE IDEA; IT IS CLOSELY RELATED TO RESEARCH IN MORE GENERAL KNOWLEDGE-SYSTEM ISSUES, AND IT COVERS A WIDE RANGE OF TECHNOLOGIES.

Boose also illustrated how personal construct methods had been applied to the problem of capturing design knowledge for a corporate memory facility of Space Station Freedom.

Real-time problem solving. The invited talks started with Chandrasekaran's presentation on knowledge acquisition for real-time problems. His work has focused on the close relation between the structure of the task and the knowledge needed for it. He brought to the presentation his perspective as a proponent of the generic-task theory (which was discussed in greater detail during the closed session).

Chandrasekaran and others have emphasized the task structure as the mediating agency in knowledge acquisition and representation. He applied that point of view to the design of knowledge systems for real-time problems, specifically that of operator guidance of a nuclear power plant. He noted that all problem solving uses models of the world; to the extent that such models cannot be guaranteed complete or correct, no problem solving process can guarantee that its solutions will be correct. In real-time systems, the problem-solving goal is to generate action sequences that will keep the environment in a desired state in the face of changes and disturbances. Since success in this enterprise cannot be guaranteed, Chandrasekaran said the best that can be done is to know how to abandon goals that are not being achieved and replace them with goals that are more likely to be achieved. Thus, all realistic real-time problems require a goal structure that supports rapid decision making about abandoning and substituting goals.

Chandrasekaran also noted that people attain successful real-time problem solving not from complex reasoning at runtime about resources or by various forms of "metareasoning," but by avoiding complex runtime reasoning altogether. This ability arises from three specific sources:

(1) Well-designed sensor and action systems, so that mapping from observations to internal states of importance, and from internal states to needed actions, is as direct as possible.

(2) A precompiled body of knowledge that helps order goals in terms of priorities and preferences, so that goals can be abandoned and modified as needed.

(3) Another body of knowledge that helps to synthesize action sequences (plans).

Chandrasekaran said this form of task analysis is useful for knowledge acquisition, since it identifies the types of knowledge to seek in the domain. (This is exactly what nuclear engineers have been doing in practice.)

The system-model-operator metaphor. Clancey discussed common languages for reuse and communication, tasks, and interaction. (His article with Monique Barbanson will appear in a future issue.) He

The *IEEE Expert* special series

The articles in this special series are based on work presented at the first Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop. This issue features Thomas Gruber's work on justification-based knowledge acquisition (pp. 65-75), which was reported in the open session.

Future issues will feature four more articles from the open session — William J. Clancey and Monique Barbanson on the system-model-operator metaphor; Atsuo Kawaguchi, Hiroshi Motoda, and Riichiro Mizoguchi on interview-based knowledge acquisition; John McDermott et al. on how to make application programming easier; and Ross Quinlan on induction from structured data — and two from the closed session — Shigenobu Kobayashi and Kotaro Nakamura on explanation-based learning, and B. Chandrasekaran on tasks and methods.

described a wide body of work, including generic tasks and role-limiting methods, in a unified, very general framework.

We can view expert systems as programs that use and construct models of some real-world system so that it can be assembled, repaired, controlled, etc. In the framework Clancey proposed for qualitative modeling of domains and tasks, the major difference between conventional computer programs and expert systems is that the models in the latter use relational networks to represent processes and structures. Control knowledge for constructing such a model consists of operators that construct a graph linking processes and structures (for example, causally, temporally, spatially, or by subtype).

Clancey claimed that every expert system contains a blackboard architecture model, whether the implementors describe it in such terms or not, and he illustrated his point with an example based on the Emycin system model. He suggested extending blackboard terminology to ease communication between knowledge engineers, and he developed a system that includes this model explicitly: Topo, an expert system for the logical topology design of computer networks. Clancey used Topo to build an experimental application that reused task operators and built a task ontology for reuse.

Clancey does not think that expert systems are models of intelligence as much as they provide a new and powerful way to model systems and processes. He proposes that "recurrent macrostructures" in such models — both of objects and processes — can be shared.

An interview architecture based on dynamic analysis. Motoda presented a joint research program whose aim is to build interviewing systems for knowledge acquisition and which has developed three systems. (His article with Atsuo Kawaguchi and Riichiro Mizoguchi will appear in a future issue.) First, the Shell for Interview Systems (SIS) is a metatool that generates specialized knowledge-acquisition tools, in the way that Protégé generates tools such as Opal. Motoda showed that it is possible to prepare a set of domain-independent and task-specific primitives (in this case, the task is interviewing itself) to describe interviewing strategies. Second, IIS-LD is an interviewing system generated by SIS for the logical design of databases. Finally, IIS-DB is an improved version of IIS-LD that includes dynamic analysis. The development of these systems showed the importance of both static analysis and dynamic analysis (testing with feedback).

Motoda's work demonstrated the usefulness of interview systems and dynamic analysis, and the feasibility of building metalevel knowledge-acquisition tools. In particular, he emphasized his system's ability to identify which knowledge is lacking or erroneous by actually solving a problem, and its ability to refine the knowledge base as it goes through a problem-solving process. His approach helps make ambiguous knowledge explicit by forcing the user to think about the way a problem is being solved, and it can handle requirements that were not thought of initially.

Motoda then proposed a general knowledge-acquisition architecture based on experience with these systems. The architecture is intended to incorporate multiple machine-learning techniques, especially those based on failure analysis. Part of this idea has been implemented as the Isak system, which was presented at the closed workshop.

Inductive knowledge acquisition from structured data. Quinlan, the originator of the Iterative Dichotomizer 3 (ID3), reviewed the progress of inductive learning systems for learning classification rules from examples, and the wide range of applications in which such systems have been used. One of the deficiencies of ID3 and related algorithms is that they are not applicable to structured data, that is, data with complex relationships to other data. He presented recent work on the First-Order Inductive Learner (FOIL) that partially circumvents this problem. By combining induction strategies — divide and conquer (as in the ID3, the Concept Learning System, and Classification and Regression Trees), and covering (as in AQ11 and CN2) — FOIL learns relations between objects with varying numbers of components. It finds a set of Prolog clauses that defines a relation in terms of itself and the other relations one at a time. There is no notion of proof (identification in the limit) as in the Model Inference System algorithm proposed by E.Y. Shapiro.[2] FOIL is solely empirical and thus presents an interesting contrast with the MIS algorithm.

FOIL showed accurate and fast results on several standard machine-learning examples, such as learning recursive relations in lists, chess endgames, and the card game Eleusis. In the chess tests, FOIL ran from 10^3 to 10^4 times faster than Duce and Cigol, and with greater accuracy. Quinlan is looking for large application problems for further testing.

Research in Japan. Kobayashi presented a jointly developed survey (the work of four authors) of Japanese knowledge-acquisition and machine learning research. Japan has clearly made a major commitment to knowledge systems, as indicated by the breadth of applications, degree of commitment, and progress reported by Japanese industry, government, and academia. One recent example of note is a project by the Advanced Software Technology and Mechatronics Research Institute of Kyoto. This joint project between academia and industry aims to build a very large engineering knowledge base that will serve as an infrastructure of expert-system technologies.

(Kobayashi's talk covered a wide territory. We have abbreviated or omitted discussions of work presented by others at the workshop or reported elsewhere.[3])

Case study. The Japanese Information Processing Development Center recently studied a number of expert-system projects in Japan. The objectives of the case study were to identify the causes of the knowledge acquisition bottleneck by analyzing real expert-system development processes, and to explore future knowledge-acquisition architectures. The case study clearly indicates that

- task analysis and identification of the problem-solving functions involved in the task are a difficult first step,
- domain models can be acquired in the form of generic prototypical libraries without waiting for the system development stage,
- no uniform formal methods have emerged as most appropriate, and
- the role of knowledge acquisition changes with the developmental stage, and different requirements must be supported at each stage in the life cycle.

Interview systems. A number of systems are being developed to support the interview process, including SIS (described earlier), Multis (a multitask interview system with task analysis function), Isak (an interview system for acquiring design knowledge using examples), and Epsilon/One, which uses an expert model. (A system that generates task-oriented expert systems is described elsewhere.[3] Multis, Isak, and Epsilon/One are described later in this article.)

Other systems include Consist (a construction support system for information structure, developed by Criepi), and Grape (groupware for acquiring, processing, and evaluating knowledge, developed by Fujitsu). Where many knowledge acquisition systems use top-down elicitation guided by the identified task type, Consist uses bottom-up interviews. It uses the KJ method (named after its inventor, Kawakita Jiro) and provides useful assistance even when the problem-solving knowledge is vague for domain experts. The Grape tool elicits expertise from multiple experts for selection problems using an analytic hierarchy process. It exploits a "what you see is what I see" interface.

Knowledge compilers. One major area of research is the development of Osaka University's Knowledge Compiler II for diagnostic expert systems. KCII uses qualitative reasoning, is quite general, and makes building a shallow knowledge base relatively easy. KCII is designed to understand the nature of deep and shallow knowledge as well as their relationship. Deep knowledge in KCII is defined as modular and objective knowledge that depends on the domain, not on the task. Task knowledge is embedded in the compiler. KCII dynamically understands which deep knowledge to use to produce shallow knowledge for a specific task. The latest results in this work include a domain- and task-specific shell, called KCII/Domain Specific Tool, which can hierarchically organize compiled rules and infer intermediate hypotheses as well as deal with devices with feedback loops.

THERE WAS SUBSTANTIAL INTEREST IN THE KNOWLEDGE ACQUISITION FRONT-END AND THE QUESTION OF HOW TO SIMPLIFY THE PROCESS OF GETTING KNOWLEDGE INTO THE MACHINE.

Mechanicot, developed at Icot, is an expert-system design tool based on knowledge compilation. The design process is formulated as a constraint satisfaction problem. Mechanicot is based on the idea that constraints described declaratively can be transformed into procedural representations using a model of the design process. It has been applied to the parametric design of gear units.

Machine learning: similarity-based learning. Osaka University's Adaptive Rule Induction System (ARIS) is an extended version of ID3, augmented by an attribute generation mechanism and generalization operators. Attributes can take numerical values. The system can coordinate with production rules written by human experts, and an integrated environment lets it amalgamate human knowledge and machine-learned knowledge. ARIS outperformed production rules written by human experts for the task of phoneme recognition in continuous speech.

Other systems include Hokkaido University's Learning System/1 and the Tokyo Institute of Technology's learning system, which performs a constructive induction. LS/1 is a domain-independent inductive learning system that repeats question-response-answer interactions with a teacher and learns the structure of the relation between questions and answers. The institute's learning system has an inverse resolution operator to invent new terms, and it learns regular sets represented by regular logic programs. It improves the man-machine interaction process needed to invent new terms.

Machine learning: explanation-based learning. Kobayashi focused much of his talk on explanation-based learning with imperfect domain theory. Imperfectness can be classified into four types: incompleteness, intractability, inconsistency, and nonoperationality. Kobayashi discussed four approaches proposed by the Tokyo Institute of Technology: augmented explanation-based learning, the explanation-based generalization version space method, explanation-based learning with metalevel domain theory, and explanation-based learning with background theory.

Augmented explanation-based learning unifies explanation- and simulation-based learning and handles plural examples simultaneously. Similarities among multiple examples can be handled using the notion of the "least explanation-based generalization." For the nonoperational theory problem, the least explanation-based generalization is more operational than usual explanation-based generalizations. Assuming that a domain theory is inconsistent if and only if some negative examples are provable as positive ones, an explanation-based generalization version space can be constructed, and the best set of macro rules (those including all positive examples and excluding all negative ones) can be searched for in this space. For the intractable theory problem, the computational load shifts from explanation-based learning to simulation-based learning to resolve the imperfectness of the domain theory. In this case, a metalevel domain theory is effective in

reducing the computational load. For the incomplete theory problem in classification, use of a background theory in addition to the domain theory is effective, where a background theory comprises commonsense knowledge such as that found in textbooks or dictionaries. Here, knowledge transfer from the background theory to a domain theory is the task of learning where the least explanation-based generalization, analogical reasoning, and case-based reasoning can be useful.

Kobayashi then discussed Osaka University's PIL (a paradigm-based inference learner), which acquires strategic knowledge of problem solving by learning macro-operators (subsequences of operator applications) from the given solution. PIL uses a heuristic called "perfect causality" to select operators. Macro-operators are generalized using explanation-based generalization. Good performance was obtained for two different problem domains: linear equations and robot planning.

Other topics included learning from partial explanation structures (from the Tokyo Institute of Technology) and "frustration-based learning" (from Hitachi[3]). The former proposes a method to specialize and generalize a set of rules in an explanation structure by composition and decomposition. The usefulness of this approach was demonstrated by application to logical circuit synthesis and English-Japanese translation.

Case-based reasoning. Case-based reasoning, which explicitly uses experts' experiences with past cases of success or failure, supplements knowledge-based reasoning, which requires complete and consistent knowledge bases.

Topics that Kobayashi discussed here included interactive modeling for case-based reasoning, cost-benefit analysis, case-based process diagnosis, and case-based information retrieval. Interactive modeling supports problem indexing and case modification and repair by interacting with users to fill out incomplete domain knowledge. (Related work presented by Kotaro Nakamura in the closed session is discussed later in this article.) The types of knowledge used to automate case-based reasoning were analyzed, with emphasis on the importance of conceptual hierarchy knowledge, causal knowledge, and heuristic knowledge in case modification and repair. The Caos process-diagnosis system uses a dynamic memory-organization system in which diagnostic cases are indexed both by the symptoms and the causal relations among symptoms. The information retrieval method is based on Carbonell's derivational analogy framework and organizes retrieval cases in a lattice structure.

How to make application programming easier. McDermott discussed work at Digital Equipment Corporation on three knowledge acquisition tools — Spark, Burn, and Firefighter — that work together to give users intelligent access to a pool of reusable mechanisms that act as building blocks to form an application.

CASE-BASED REASONING, WHICH USES PAST SUCCESSES OR FAILURES, SUPPLEMENTS KNOWLEDGE-BASED REASONING, WHICH REQUIRES COMPLETE AND CONSISTENT KNOWLEDGE BASES.

This work is based on lessons learned from work on such systems as Teiresias, MDX, Neomycin, Roget, EES, More, Mole, Salt, Kanck, Leap, Cyc, Protégé, ASK, Rime, Accord, and Taql. These lessons include:

- Representing control knowledge abstractly illuminates the possible uses for pieces of knowledge (Neomycin).
- A problem-solving method can be exploited to guide the elicitation and encoding of knowledge (Roget).
- A problem-solving method imposes roles on the knowledge it uses, and the existence of these roles can be used to guide knowledge acquisition (More).
- Control knowledge comes in several varieties, and distinguishing among the varieties substantially enhances a program's intelligibility (Rime).
- Given an appropriate computational model, an application program can be created from a task-level description (Taql).

Spark takes some inspiration from Chandrasekaran's generic tasks and from James Bennet's Roget system. Spark contains a description of the domain ontology and mechanisms. It interviews a developer, and based on what the developer says about the information available for and the results desired from a task, it selects from a library of predefined computational mechanisms one or more mechanisms that can collectively produce the desired results. It then configures those mechanisms into a problem-solving method.

Burn performs knowledge elicitation and encoding. It is essentially a manager of knowledge acquisition tools that have been developed by many researchers. It retrieves, from a library of tools, the one associated with each of the mechanisms selected by Spark. It uses those tools to elicit expertise from the developers and produce an application program by encoding that expertise in a way that allows it to be used by the problem-solving method previously configured by Spark.

Firefighter runs the application program and gives the user feedback while the system is being developed. Whenever the application program is used by one of the developers, Firefighter asks whether the program has performed correctly; if it has not, Firefighter invokes either Burn or Spark to add knowledge to the program or modify the problem-solving method. It is a relatively simple "diagnostician" of problem-solving methods and knowledge, and it uses a variety of techniques to "fix" the knowledge or methods. McDermott illustrated the system in the domain of computer sizing estimation.

Justification-based knowledge acquisition. Gruber described two knowledge acquisition problems that require the domain expert to formalize and operationalize knowledge that is currently expressed only informally:

(1) Acquiring strategy. How do we elicit strategic knowledge from experts without requiring them to design procedures?

(2) Device teleology. How do we elicit explanations of a device's "purpose"?

A technique called justification-based knowledge acquisition, developed for the first problem in the context of medical diagnosis and therapy, is embodied in a program called ASK (for "acquiring strategic knowledge"; see Gruber's article on pp. 65-75). The program asks an expert to

critique the strategic actions of a running expert system and to provide justifications for why the system should have chosen different actions in similar situations. ASK provides a sophisticated interface through which the expert can choose an appropriate action together with relevant features of the current situation. After being told "what" and "why," ASK generalizes this advice into rules that represent strategic knowledge.

The second problem was discussed in terms of work in progress on a device-modeling environment called DME, a methodology for acquiring the design rationale for electromechanical devices. DME contains model-based explanations of how things work and provides intelligent simulation support. The user justifies a design decision by demonstrating the behaviors enabled or prevented by the designed structure using a qualitative simulation, comparing alternative designs in the same simulation context. Features that justify one design decision over another are states of a running computer model of the process — in this case, a simulation instead of a diagnosis.

Gruber noticed the similarity between the methods in ASK and DME and concluded that justification-based knowledge acquisition covers broader problems and is very general. The strength of this approach lies in how it divides the knowledge acquisition load between the user and the machine. The machine provides the computational model and the user provides access to the external world by selecting what is relevant among the possibilities afforded by the computational model, as well as by defining new vocabulary based on existing terms. Since the knowledge is stated in the machine's language, and the machine must formulate it in the context of its task, the acquired knowledge is always operational.

Integrating knowledge acquisition and performance systems. Rappaport emphasized the importance of dynamic feedback (testing) and discussed the implications of reusing pieces of knowledge-based systems. He said there is a basic trade-off between flexibility and complexity when deciding how large the chunks should be when breaking systems into components for reuse. He suggested an architecture composed of reusable chunks from knowledge-based system tools as well as other conventional tools (such as user interfaces and database management systems).

The closed session

The notion of knowledge reuse, raised during the open session, also dominated many of the presentations and discussions in the closed session. Many systems try to reuse knowledge at different levels (task knowledge, problem-solving knowledge, task ontologies) to gain leverage during knowledge acquisition. Discussions centered around two main topics: the nature of knowledge, and knowledge reuse. Many metaphors were used to describe knowledge and knowledge acquisition, including "chicken soup," "filtering process," and "bottleneck" metaphors. All of them generated controversy.

THE NOTION OF KNOWLEDGE REUSE, RAISED DURING THE OPEN SESSION, ALSO DOMINATED MANY OF THE PRESENTATIONS AND DISCUSSIONS IN THE CLOSED SESSION.

The methods used covered a wide spectrum, but many of them combined deductive and inductive approaches. Deductive approaches were used to guide searches by taking advantage of domain or background knowledge in acquiring knowledge by inductive inference. Rule induction for acquiring classification knowledge focused on rule analysis and refinement. Interviewing was also guided by domain knowledge and coupled with deductive learning. Generic models described at the knowledge level were discussed extensively in interview and other systems. Hierarchical knowledge compilation and sophisticated hypertext-based interfaces also attracted attention.

About 40 people attended the closed session, including the invited speakers from the open meeting. It started with opening remarks by Motoda and Boose. Eighteen papers were presented, including an invited talk by Chandrasekaran. The workshop was rounded off by Boose's demonstration of Aquinas attempting to acquire knowledge useful for a Japan tour from all the participants.

Tasks, methods, and knowledge. Chandrasekaran is an advocate of the concept of generic tasks, which are now widely accepted as building blocks of expert systems. His talk was based on his research experiences and reflections over the past several years. He used design problem solving as an example task around which to organize his thinking on the relationship between tasks, methods, and knowledge. He emphasized that the key to understanding real-world design is not in any uniform general algorithm for design — some sort of design from first principles — but in how the design task is decomposed into subtasks and how computationally appropriate methods are developed and associated with the subtasks.

In general, a method is a proposal about how to organize information processing to achieve a goal. Methods typically set up additional subtasks, for each of which a number of methods can be identified. Each method is characterized by the types of knowledge and strategies needed to support it. For a complex task such as design, we could develop a task structure that organizes alternative methods for each task and identifies the subtasks for each method. The task structure is thus a recursive analytic structure. This recursive representation proceeds until it gets to a feasible problem space where executable methods are available.

Chandrasekaran used the Soar architecture to explore this idea. He said that the most common top-level family of methods for design is based on the "propose-critique-modify" approach, which proposes partial or complete design solutions, verifies the proposed solutions, critiques the proposals by identifying any causes of failure, and modifies the proposals to satisfy goals. Design proposal methods include problem decomposition/solution recomposition, retrieval of case bases, and constraint satisfaction. Because of the multiplicity of possible methods and subtasks for a task, a task-specific architecture for design is not likely to be complete. Therefore, Chandrasekaran concluded that

Other knowledge acquisition workshops

The first Knowledge Acquisition for Knowledge-Based Systems Workshop was held in Banff, Canada, in 1986. Workshops have been held in Banff each year since. In 1987 the first European workshop in the series was held in London and Reading, England. Since then, yearly workshops have also been held in Europe (in Bonn, Paris, Amsterdam, and Crieff, Scotland). The 1990 Japanese Knowledge Acquisition Workshop marks the first Pacific Rim workshop in the series. Planned future workshops include:

AKAW 91: The First Australian Knowledge Acquisition for Knowledge-Based Systems Workshop (second in the Pacific Rim series), Sydney, August 20-23, 1991. Ross Quinlan is the local organizer.

KAW 91: The Sixth Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, October 6-9, 1991. John Boose and Brian Gaines are the organizers.

EKAW 92: The Sixth European Knowledge Acquisition for Knowledge-Based Systems Workshop, Heidelberg, Germany, May 18-22, 1992. Marc Linster, Franz Schmalhofer, and Thomas Wetter are the local organizers.

JKAW 92: The Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop, Kobe and Saitama, November 9-13, 1992. Hiroshi Motoda and Riichiro Mizoguchi are the local organizers.

Tentative dates for future Banff workshops are October 11-16, 1992, October 10-15, 1993, and October 9-14, 1994.

attempting to build monolithic task-specific architectures is not the right approach; instead, we should capture the target task as a composite of appropriate methods combined in a domain-specific manner. The Soar architecture also makes runtime configuration of methods possible.

Rule induction. K.S. Leung presented an automatic knowledge acquisition and refinement system Arkas, which acquires classification rules by induction from training examples and generalizes them by dropping insignificant conditions as a primary refinement. It then progressively refines the rules to improve their performance by executing the knowledge base using known cases that differ from the training examples. Arkas' induction algorithm employs a covering method. It uses conditional probability as a bias to select the best attribute/value pairs. The types of attributes it can handle are nominal, linear, and structural, and it uses different methods to evaluate the best pair of each; the pair with the largest conditional probability is selected as the real best pair. Checking the significance of a condition in a rule is reduced to the problem of statistically testing the dependency of the condition on the classification. The whole algorithm was tested for the data available in the open literature, and the results compared favorably with ID3. One attendee of the presentation pointed out that predictive accuracy must be discussed in terms of unseen data and not the training data.

Integrating inductive and explanation-based learning. An induction algorithm like ID3 outputs a reasonable collection of knowledge, but because the algorithm does not know anything about the domain, the results are sometimes not good enough from the expert's point of view. To address this problem, Katsuhiko Tsujino's system aims to integrate inductive and explanation-based learning. The basic idea of the system is to use domain knowledge to guide the inductive inference. It first acquires classification knowledge in the form of a decision tree by induction using the ID3 algorithm. It analyzes the results and processes with domain- and task-specific knowledge to detect improper conditions and mismatching between the theory and the results. It then asks suggestive questions to eliminate the improprieties and acquires new domain knowledge and various information for the next induction cycle.

Improprieties can be structural or semantic. The former is task-specific (classification in this case) and domain-independent, and the latter is domain-dependent. Structural improprieties are detected as generalization failures or unreliable conditions, and the remedies can be generated in a general way. Semantic improprieties are detected as the inability to explain results and are further classified into four categories, each of which has its own remedy method. Question strategies are prepared based on the current combination of improprieties. This method was applied to a simplified version of Tetris (a brick-piling game) and worked as expected.

Similarity-based learning and conceptual clustering. Mitchell Medow's Hamatmid is a phased system for heuristic classification problems. It tries to simulate the human development of expertise: assimilating textbook knowledge, solving simple problems with a teacher, eventually handling more difficult cases with feedback about the correct outcome, and finally developing expertise on its own by compiling the acquired knowledge.

Medow elaborated on Novice, the first stage of Hamatmid, which develops a web of solution classes based on background knowledge and extracts problem-solving rules to traverse this network. Novice uses similarity-based learning and conceptual-clustering techniques with the prototypical cases described in textbooks. Each prototype is presented as a set of attributes, each with different values. Novice builds multi-dimensional tangled hierarchies (networks) from the prototypes, learns new structural knowledge, and organizes itself for its diagnostic task, from which it extracts four kinds of classification rules: pathognomonic, exclusionary, splitting attribute, and multiple-level rules.

To compare Novices' performance with Mycin's, Medow gave Novice prototypes for bacteremia (blood infections) and background knowledge from a standard medical text, augmented with other teaching material. Novice included all of Mycin's suggestions, plus a few additional ones that resulted from the excessive generality of Novice's rules. It appears that this method, although its main focus is on medical diagnosis, is applicable to any other domain where the task is classification, the solution classes are presented as prototypes, additional background knowledge exists to supplement these solutions, and a well-developed repository of educational material is available.

Refining classification knowledge. Kenichi Hagiwara presented a method to refine classification knowledge. He devised a representation called the "choice component with include-option" for the conditional part of classification rules. This

scheme begins with a set of findings (observations) that support a particular classification. A certain subset of the findings is assumed to be particularly important. The technique specifies that at least a minimum number (n) of the important findings be true in addition to at least a minimum number (m) of all findings. This expression is appropriate when the relation between findings and the classification target is not clear and the knowledge base needs refinement. The system searches for the numbers n and m in a base finding space expressed as a hypercube, using both positive and negative examples according to a certain algorithm. If an unimportant finding is included, m will be small so that it will not be included in the important-finding list. If an important finding is missing, a conflict arises between the positive and negative examples, which is detected during search. Since the algorithm works on a partially ordered subset of the cube, the modification is monotonic. The current algorithm discards both examples if they conflict on the same given set of the base findings, but an attendee of the presentation noted that the system should be able to find which one to discard.

Rule-induction algorithms. Geoffrey Webb compared two different algorithms for the problem of rule induction for classification. A data-driven machine-learning system tries to maximize the classification procedure evaluation criteria, which is usually the weighted sum of two measures: the classification accuracy of the training set, and the simplicity of the classification procedure. Previous implementations of the AQ algorithm (a "covering" method) have used rule-optimization search strategies to develop optimal classification procedures by generating successive characteristic descriptions, each of which maximize the given criteria. This type of strategy is subject to the flaw that individual characteristic descriptions that rate highly will not necessarily form a classification procedure that rates highly. An alternative is the theory-optimization search strategy, which generates successive complete classification procedures and selects those that maximize the evaluation criteria.

These two algorithms were compared using simple evaluation criteria for the domain of diagnosis of immunoglobulin a nephropathy diseases. Simplicity is measured by the number of characteristic descriptions in the classification procedures. The results indicate that theory optimization outperformed rule optimization at least for a small set of data with a high level of missing values.

Hypertext. Padraig Cunningham's combined approach to text retrieval uses concept networks linked to hypertext. The process of knowledge acquisition for text retrieval tries to produce conceptual structures representing the information contained in the text. Cunningham used the metaphor of "chicken-and-dumpling soup": The dumplings are factual knowledge, the soup is tacit knowledge, and the expert system is a colander that can transfer only factual knowledge. It is the user who supplies the tacit knowledge that makes the whole process work.

TWO BIG PROBLEMS IN UNDERSTANDING A TEXT SENTENCE ARE RESOLVING INFERENCES AND UNDERSTANDING METAPHORS, BOTH OF WHICH REQUIRE TACIT KNOWLEDGE.

Two big problems in understanding a text sentence are resolving inferences and understanding metaphors, both of which require tacit knowledge. Cunningham's approach doesn't try to create concepts for difficult passages; instead, it gives a user direct access to the text, letting him use his own tacit knowledge to understand the passages. A hypertext network provides the text of all documents in the system. Links from the concept network to the hypertext, and reference links within the hypertext, bring the user to the passage from which the concept was extracted, as well as to related passages. The concept network and the reference links in the hypertext are created in parallel when the hypertext is generated from a source document.

Uwe Schreiweis also presented a hypertext-based knowledge-engineering support system, called Concorde. This research prototype is suited for active hypertext applications, such as decision support, authoring, and knowledge structuring. It has global hypertext and local contexts. The global hypertext holds information common to all users, whereas the local context supports an individual user's view of the global hypertext. A local context contains a subset of the nodes and links copied from the global hypertext, and adds to them its own nodes and links. Knowledge modeling is performed in the local context. Links in the global hypertext (and thus those in the local context copied from the global hypertext) are typed and express various constraints. Taking advantage of this system's properties, an integrated approach becomes possible for collecting knowledge, managing sources of this knowledge, supporting its structuring, and maintaining it. Schreiweis is convinced that this approach represents an important step toward the integration of databases, expert systems, and hypermedia systems.

Explanation-based learning and case-based reasoning. Kotaro Nakamura used explanation-based learning to acquire procedures (rules) for repairing machine problems. (His article with Shigenobu Kobayashi will appear in a future issue.) He used previous repair cases (called adjustment cases) as examples, the machine's causal model as a domain theory, and observability and strategic importance as operationality criteria. Many of the cases were available from existing documentation, but some had to be extracted directly from domain experts, which was not too difficult because experts remember a case as a kind of episode.

In Nakamura's approach, the explanation structure is generated by using the causal model to prove that a given case is an instance of the adjustment goal. Next, the causal paths are simplified by eliminating paths that are inconsistent with the case's faults. Unobservable nodes and those with a low value for the operational criteria are then removed from the paths. The simplification here corresponds to the generalization (and equally to the reduction) of the explanation structure. The resulting path forms a primitive rule. Redundant rules are removed and disjunctions of the rules are taken to form an appropriate level of granularity. The system has been implemented and applied to a cigarette-making machine.

It generated 79 rules using 50 actual adjustment cases, among which 44 rules were new, 19 were refinements of existing knowledge, and 16 were justifications of existing knowledge.

Knowledge acquisition can also be coupled with machine-learning techniques. Riichiro Mizoguchi presented Isak, an interview system for acquiring design knowledge for oil-hydraulic circuits. Isak is composed of two major modules: understanding-based learning — a module that compiles design knowledge by understanding examples of design drawings — and understanding-based interviewing — a module, activated when understanding-based learning fails to understand an example, to acquire design know-how or missing domain theory that contributes to improving the performance of understanding-based learning.

Isak has some interviewing strategies as well as initial domain knowledge. Given a design example, the system tries to explain why the example satisfies the design specification, and it identifies which components are necessary to satisfy the specification. Symbolic simulation is performed to obtain the description of the circuit's behaviors, and the proof is made, from which a piece of compiled design knowledge that maps function to attribute is extracted with the form: if {specification} then {use the component in this structure}. If the proof fails, understanding-based interviewing initiates interviewing to acquire three kinds of knowledge, according to the type of failure: hidden requirement specifications, additional domain knowledge, and design know-how that is difficult to explain in terms of the domain theory. Knowledge acquired via understanding-based interviewing is used for understanding future examples in understanding-based learning, so the scope of the system's understanding expands as the interview proceeds.

Having the domain theory in advance helps reduce the number of questions raised by the interview system. A key idea in Isak is that design knowledge obtained through explanation-based learning is a specialization of task-independent domain theory that is made operational, rather than a generalization of an example.

Interviewing. Koichi Hori presented a method to assist articulation of the mental world. In contrast with the many knowledge acquisition methods that build structures from predefined primitives, the main focus of Hori's work is building the initial set of symbols. The work is based on the assumption that there is no one concrete set of knowledge in any domain, but rather that an expert's mind has only tacit and nebulous materials, which cannot be always verbalized, for building knowledge. Hori's proposed method reflects the mental world indirectly by placing words in a two-dimensional Euclidean space, where any relation between words is represented by physical closeness. The world of the knowledge representation language was also modeled as a topological structure that shows semantic relations based on a reasoning system.

HORI'S WORK IS BASED ON THE ASSUMPTION THAT AN EXPERT'S MIND HAS ONLY TACIT AND NEBULOUS MATERIALS, WHICH CANNOT BE ALWAYS VERBALIZED, FOR BUILDING KNOWLEDGE.

Topology is flexible and powerful enough as a mathematical tool to reflect any structure of concepts. It helps find new concepts in empty places and makes it easy to look at the world from different views. Hori has implemented this idea as a knowledge acquisition tool in an application designed to articulate expertise regarding a feedback control system, and he has shown that it is flexible and effective in structuring human ideas in this application. Some attendees felt, however, that the approach should be more scientific and technologically persuasive.

Automated versus manual knowledge acquisition. Francesco Giordana described Enigma, a program capable of learning structured knowledge from examples and a domain theory, using an integrated inductive and deductive paradigm. Enigma's main objective is to explore the possibility of automatically acquiring diagnostic rules by induction that are understandable in light of preexisting domain knowledge and usable in real applications. Enigma receives as input a set of learning examples and a body of background knowledge represented by a set of horn clauses, and it produces a structured knowledge base of classification rules. It produces a graph of rules by searching in a space of rules through a specialization process, starting from the most general formula and generating more and more specific formulas until it discovers classification rules. This process is guided by statistical heuristics that trade off consistency and completeness criteria, and by the background knowledge supplied at the beginning. The results are compared with those obtained by a hand-coded expert system in which the knowledge base was acquired using the traditional method of expert interviews.

Enigma was put to the task of diagnosing motor pumps using data derived from the Fourier analysis of vibration signals. The comparison indicated that Enigma's performance is systematically better than the that of the manually developed expert system. Giordana's conclusion was that, even leaving room for improvements, automatic acquisition is a viable approach to expert-system construction from the point of view of both attainable performance and reduced development time and cost.

Integrating interviewing and induction. Kazuhiro Taki presented a proposal-guided knowledge-acquisition support system, Epsilon/One. His basic idea is that the elicitation of knowledge by induction can be made tractable by limiting the representation form and allowing the acquired knowledge to be edited by an expert. Epsilon/One has an expert model consisting of a set of operations that represents small tasks in a human expert's problem-solving process.

Taki identified seven operations by analyzing real knowledge bases built for diagnosis problems in production rules: classification, selection, input, output, ordering, combination, and translation. Epsilon/One elicits a sequence of operations by asking an expert what kind of operation is performed before and after a specific operation (the "pre-post method") and then eliciting knowledge about operations by induction according to the type of operation. For each type identified, the system extracts the criteria by analyzing the operation's

input and output, generalizes the induced criteria, and presents them to the expert. During the analysis it can also ask for comments. The expert then corrects the proposal, if necessary, and requests the next set of results.

Taki described the procedures to induce the criteria for the major operation types, and he showed an example of bearing selection, ordering, and their value replacement taken from a machine-tool design problem. Given 20 types of bearings as input and five selected types with altered attribute values as output, Epsilon/One successfully induced the criteria for selection, ordering, and replacement.

Task-analysis interviewing using a problem-solving method. Yuri Tijerino reported on Multis, an interview system that is not restricted to eliciting domain knowledge for a single task. Multis builds a model of expert problem solving with a network of highly generalized cognitive primitives, designated as generic tasks and maps that network to a combination of building blocks. The generic tasks are defined in terms of generic vocabularies — task-dependent lexica independent of domain. These generic tasks differ from Chandrasekaran's in that they must be combined to obtain executable building blocks and they are used to model human problem solving at the cognitive level. Tijerino has identified 19 generic tasks for scheduling.

Multis first roughly identifies the task the domain expert performs and then models it by combining appropriate generic tasks. The system uses case data to avoid searching (inefficiently) for all possible combinations of the generic tasks. The case-based reasoner chooses the closest case with a cognitive structure (generic tasks) and a search-control structure (building blocks) that can be modified to fit the current problem. The identified generic tasks and building blocks have structures that guide the elicitation of task-implementation and domain knowledge, based on which an interview is made.

Acquiring knowledge at different levels of abstraction. It is important for a problem solver to be able to use knowledge described at different levels of detail. Kenichi Yoshida's method constructs a hierarchical knowledge base in which the knowledge is expressed on multiple levels with different abstractions that are treated as approximations in the functional description. Different representations are used for each level, together with different domain models. For example, a carry chain circuit (part of an adder) is described at four different levels: analog, switch, gate, and circuit level. In moving up one level, some assumptions must be made (such as neglecting some variables or assuming a prompt change) and a logically correct transformation must be made. When considering the two adjacent levels, the higher-level knowledge can be regarded as the specification, and the lower-level knowledge as the implementation. This is recursive; the description of some knowledge level can be regarded as a set of specifications of the lower level and also as the implementation of an upper level. The system tries to explain the (higher level) specification using the (lower level) implementation by finding the necessary approximations, that is, by reforming a theory. The identified approximations are also stored in the hierarchical knowledge base. This system is being applied to describe basic electric circuits, such as four-bit adders.

WITH ACCESS TO REMOTE DATABASES, THE NEED ARISES TO EXTRACT RELEVANT DATA AND TO COMBINE THE PARTIAL RESULTS OF INDIVIDUAL QUERIES INTO A SINGLE ANSWER.

Miscellaneous. Ahmad Barfoursh talked about knowledge assimilation when retrieving information from distributed databases. With advances in networking capabilities that allow access to remote databases — particularly databases that provide a large amount of information about similar topics — the need arises to extract relevant data from the databases' responses and to correctly combine the partial results of individual queries into a single answer.

Barfoursh's system first elicits the retrieved text by performing a lexical analysis and parsing it to transform into a standardized frame-based representation. The system then assimilates this into the existing knowledge base. The assimilation mechanism examines incoming data to see

- if it is consistent with, or can be deduced from, the current knowledge base (existence checking);
- if it is correct if it differs from identical knowledge in the knowledge base or from the integrity constraints (correctness checking); and
- if there is any missing information, based on the integrity constraints and the current knowledge base (completeness checking).

The assimilation algorithm is specified as rules in Prolog, and it is similar to those developed at Icot. Barfoursh applied this method to on-line library inquiry. What makes this approach practical is that only facts need be treated as assimilation targets. This method lets users add new knowledge to the knowledge base, and it can also obtain, evaluate, and prepare knowledge from the database to add to the knowledge base.

Integrating knowledge acquisition with a development methodology. Dieter Wenger presented a development methodology for knowledge-based applications called Ema. Knowledge acquisition is driven in Ema by the required results, by consistency checks of the knowledge model, and by verification and validation of the prototype. A knowledge-based application is seen as an active partner for the user and as a system consisting of active autonomous cooperating agents reacting to events and bringing the system to a new consistent state. Agents use both declarative (information-based, built by concepts and attributes) and operative (function-based) knowledge. The former builds the knowledge model, and the latter makes the application executable.

The knowledge model is application specific and is constrained by the metamodel, which defines the information in an object-oriented way and is application independent. Since the knowledge model is not a model of the domain, it must be provided with functions that facilitate the efficient acquisition of domain knowledge and prototype generation. The declarative

knowledge an agent requires includes activation, input, constraint, and process context knowledge. The behavior of agents is displayed as agent views, and necessary knowledge is identified by analyzing the results. The consistency checks are triggered automatically by certain states of the information kept by Ema. This methodology was built with its own help, since the Ema knowledge model is also its metamodel, and Ema is developed as a knowledge-based application that is active and executable.

Recursive learning. Morihiko Tajima's Recursive Learning System learns procedural knowledge. Like Heurisko's system, RLS can modify metaknowledge (knowledge to modify knowledge) as well as the target procedural knowledge. RLS's most important component is a metaknowledge generator. RLS observes a rule's execution, and if the results are not good, it generates metaknowledge to modify the procedural knowledge. Similarly, if it observes that the generated metaknowledge is not performing as expected, RLS generates metametaknowledge to modify the metaknowledge. The modification is based on a general learning technique and on learning that RLS has done in the past.

Both knowledge to be modified and knowledge to modify something are described in a uniform manner, so the same learning algorithm is applied to both kinds. Knowledge is expressed as a collection of knowledge pieces, each of which has a distinct type. A knowledge piece can have knowledge subpieces. Thus, RLS has three kinds of recursiveness: metaknowledge modifies knowledge, knowledge pieces have knowledge subpieces, and the system generates metaknowledge for target knowledge.

Tajima applied RLS to learn the rules of the game Othello and showed that RLS worked well. The learning process is complicated, but the whole process is based on the general structure of RLS, which itself has very simple execution and generation mechanisms.

THE PRESENTATIONS AT JKAW addressed many important areas of knowledge acquisition, but several issues remain that were addressed indirectly, only slightly, or not at all.

Kads. The main goal of the Kads (knowledge acquisition and structuring) project, for example, is to build a methodology for knowledge-based system construction and a knowledge engineer's workbench.[4] Developed under the ESPRIT project, the methodology is based on a model-driven approach in which a conceptual model of human experts plays a crucial role. Similar ideas can be found in Chandrasekaran's generic tasks, McDermott's Spark/Burn/Firefighter, Clancey's system-model-operator metaphor, and Mizoguchi's Multis. Although their relations have not been fully discussed yet, they will provide an interesting issue in future knowledge acquisition research.

SEVERAL PEOPLE DEALT WITH THE INTEGRATION OF LEARNING AND KNOWLEDGE ACQUISITION, BUT CLOSER INTERACTION IS NEEDED BETWEEN RESEARCHERS IN THE TWO COMMUNITIES.

Validation and verification. As knowledge-based systems have grown and been used more widely, validation and verification of knowledge bases has become a key issue for expert systems to be used in real-world applications requiring high reliability. V&V has been extensively discussed in the software-engineering community, and several techniques have been developed. Expert systems have different characteristics from most software and so require different V&V methodologies. However, there are some similarities. Sophisticated V&V techniques should be discussed in more detail, and we should take advantage of the results already obtained in the software-engineering community.

Combining knowledge acquisition and machine learning. The primary objective of knowledge-acquisition research is to develop techniques for making expert systems easier to build and to make them more explainable, robust, and intelligent. This has given the field an "engineering flavor." It is generally agreed that knowledge already exists in the world, and that the problem is to elicit it and make it operational. Machine learning, on the other hand, is viewed more as an activity to transform knowledge that is already operational to a more effective form or to create new knowledge, mainly from data. This area has a more academic flavor.

The research presented at JKAW clearly indicated that knowledge-acquisition methods are not as homogeneous and easily characterized as genetic algorithms. Many of them use methods developed in the machine-learning community. Both areas share a common goal: to build a knowledge base that is useful for some problem-solving activity. Several presentations dealt with the integration of learning and knowledge acquisition, but more efforts are needed to attain closer interaction between researchers in the two communities. Amalgamating the two will be one of the most critical issues in the field.

Acknowledgment

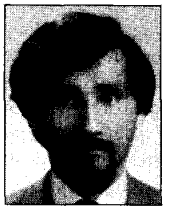
The authors are indebted to B. Chandrasekaran for his contributions to this article as well as his critical comments as a participant of the workshop and as editor-in-chief of *IEEE Expert*.

References

1. *Proc. First Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop (JKAW 90)*, H. Motoda et al., eds.; in North America: IOS Press, Burkem, Va.; elsewhere: IOS Press, Amsterdam; 1990.
2. E.Y. Shapiro, *Algorithmic Program Debugging*, MIT Press, Cambridge, Mass., 1982.
3. H. Motoda, "The Current Status of Expert System Development and Related Technologies in Japan," *IEEE Expert*, Vol. 5, No. 4, Aug. 1990, pp. 3-11.
4. J. Breuker et al. "Model-Driven Knowledge Acquisition: Interpretation Model, Deliverable Task A1, Esprit Project 1098," Tech. Report 87, 1988.

Hiroshi Motoda is a chief research scientist at Hitachi's Advanced Research Laboratory and heads its AI group. His research focuses on machine learning, knowledge acquisition, knowledge compilation, and qualitative reasoning. He received his BS, MS, and PhD in nuclear engineering from the University of Tokyo. He is an editorial board member of *IEEE Expert*, the *Journal of Knowledge Acquisition*, and the *Journal of Computer Science*, and a member of the board of trustees of the Japanese Society for Artificial Intelligence. He is also a member of the IEEE Computer Society, AAAI, JSAI, JSSST, IPSJ, and AESJ.

His address is Advanced Research Laboratory, Hitachi, Hatoyama, Saitama 350-03, Japan; e-mail, motoda@harl.hitachi.co.jp



Riichiro Mizoguchi is a professor in the Research Department of Electronics at the Institute of Scientific and Industrial Research of Osaka University. His research interests include nonparametric data analysis, speech understanding, knowledge engineering, and intelligent tutoring systems. He received his BS, MS, and PhD from Osaka University in 1972, 1974, and 1977, respectively. He is a member of IEEE; AAAI; JSAI; IPSJ; the Institute of Electronics, Information, and Communication Engineers; the Acoustic Society of Japan; and the Japan Society for CAI.

His address is Institute of Scientific and Industrial Research, Osaka University, 8-1 Mihogaoka, Ibarki, Osaka 567, Japan; e-mail, miz@ei.sanken.osaka-u.ac.jp

John Boose is principal scientist in the area of knowledge acquisition at the Boeing Advanced Technology Center, and a visiting research associate at the University of Calgary's Knowledge Science Institute. He is currently applying knowledge-acquisition techniques to group decision support systems and to the capture of design knowledge. Boose is coeditor of the journal *Knowledge Acquisition*, and the book series titled *Knowledge-Based Systems*.

His address is Advanced Technology Center, Boeing Computer Services, Bldg. 33.07, 2760 160th Ave. SE, Bellevue, WA 98008; e-mail, john@atc.boeing.com

Brian Gaines' photo and biography were not available when this issue went to press. His address is Department of Computer Science, University of Calgary, 2500 University Dr. NW, Calgary, Alberta T2N 1N4, Canada; e-mail, gaines@cpsc.ucalgary.ca

Call for Articles

IEEE EXPERT
Intelligent Systems and their Applications

Special Series on Artificial Intelligence in Text-Based Information Systems

This special series will feature articles reporting the application of artificial-intelligence techniques to the design and development of text-based information systems. Text-based information systems are characterized either by databases that contain predominantly natural-language text (such as office systems and bibliographic applications), or by the central role that text plays in the retrieval of other types of information (such as in multimedia applications that store images, video, graphics, and so on). Effective retrieval in a text-based system is difficult because much of the information is unstructured, and because the semantics of text is not easy to define. AI techniques could improve many aspects of such systems, from the acquisition, analysis, representation, and organization of information, to the retrieval, evaluation, and refinement of information in response to user queries.

Articles submitted for this series should clearly explain the approaches and techniques used and the impact of these techniques on the information system's performance. Articles that include reports of performance evaluation are preferred. We are particularly interested in descriptions of systems used in industry or systems that use databases developed from real applications.

Audience

IEEE Expert takes the ideas and tools that come out of research and development and transfers them to the user community. Clear, not overly formal, writing is essential. The magazine's readers use, develop, manage, research, and purchase databases, expert systems, and AI tools. They want to learn about tools, techniques, concepts, and systems with potential for real-world applications. Conceptual or theoretical articles are welcome, provided they clarify or present ideas of potential importance to applications.

Special Series

A special series is a collection of articles united by a theme; it is just like a special issue except that the articles appear over several issues.

Submissions

All manuscripts submitted for publication should be original. Articles published in other journals will not be considered. Submissions should run 8 to 10 journal pages, with 10 to 12 references. Four copies of the full manuscript should be submitted to either of the guest editors before October 25, 1991. Manuscripts will be reviewed by at least three reviewers. For full submission guidelines, please contact one of the guest editors:

Dik L. Lee
Dept. of Computer and Information Science
Ohio State University
2036 Neil Ave.
Columbus, Ohio 43210-1277
dlee@cis.ohio-state.edu

W. Bruce Croft
Dept. of Computer Science
University of Massachusetts
Lederle Graduate Research Center
Amherst, MA 01003
croft@cs.umass.edu