# Knowledge and Data Engineering

## I. Introduction

WELCOME to the wonderful world of *Knowledge and Data Engineering*. We are grateful that you have become the founding (first) readers of our new journal. We hope that you will enjoy this first issue. As always, we will be delighted to receive your comments and suggestions so that we may better serve you.

This quarterly TRANSACTIONS aims to provide an international forum to publish results on the research, design, and development of knowledge and data engineering methodologies, strategies, and systems. It is intended to fill a gap in the engineering aspects of the acquisition and management of knowledge and data and provide a unified channel to publish results in this area. The TRANSACTIONS is interdisciplinary and covers areas including, but not limited to, computer science, artificial intelligence, electrical engineering, computer engineering, and cognitive science.

According to Webster's dictionary [8], data refer to numerical information suitable for computer processing, while knowledge refers to the sum or range of what has been perceived, discovered, or learned. Knowledge can be considered data at a high level of abstraction and can be processed by a computer when it is represented as data. The distinction between these two concepts in terms of computer processing is usually vague [1], [10]. For example, a statement that data engineering is a growing field is a piece of knowledge. This statement has to be substantiated by facts (or data) or by algorithms (or programs) that can generate the supporting evidence. Since the amount of data necessary to characterize a piece of knowledge can be infinite in size, so the meaning of a piece of knowledge can be imprecise or uncertain. In general, knowledge can be considered as a compact and sometimes imprecise way of representing a body of data.

The subjects covered in this TRANSACTIONS are on data and knowledge engineering, which collectively refers to the methods, algorithms, and systems for the design, utilization, and maintenance of data and knowledge. This involves: 1) methods for automated acquisition and learning of new data and knowledge; 2) modeling, design, access, control, and evaluation of data and knowledge engineering systems; 3) representation, language, and architectural supports; and 4) deployment, evolution, maintenance, and standardization of data and knowledge engineering systems with existing and emerging technologies. In short, knowledge and data engineering can be considered as studies related to computer-aided management of information, data, and knowledge. The problems involved in this area are continuously evolving, as new applications arise and emerging technologies become mature. As a result, the scope covered by this journal is flexible and will change with time.

Different degrees of abstraction of a given problem in knowledge and data engineering may be required, depending on the complexity of the problem involved. Some problems require less abstraction and can therefore be implemented easily in hardware/software. An example of this type of problem is the design of a hardware selection unit for a relational database. Other problems are more complex and require a hierarchy of abstraction to simpler problems before they can be solved. These simpler problems by themselves may not represent realistic, efficient, or realizable implementations. An example is a distributed database that can handle concurrent accesses and updates. The concurrency control problem is extremely complex if all characteristics of the physical distributed system and user behavior have to be considered. To solve this problem, a simplified model of the communication network, processors, and user behavior has to be assumed. A concurrency control algorithm is then developed for the simplified model, and the algorithm is mapped onto the physical distributed system. Since simplifying assumptions have been made in developing the algorithm, the algorithm actually implemented on the physical system may not be totally efficient.

Fig. 1 shows a conceptual view of problems in knowledge and data engineering and the relationship to applications and technologies. Examples of problems with varying degrees of abstraction are also illustrated in the figure.

The early years of computing research were dominated by information processing. Loosely speaking, information refers to bits that are stored in computer memories. These bits include data, software, and knowledge represented in data and software. In von Neumann computers, for example, data and programs are stored in the same memory areas. Therefore, data can sometimes be interpreted as programs, while programs are sometimes considered as data. Since knowledge can be treated as a general class of data, its characterization may be uncertain or imprecise. Knowledge can be represented in computers as data or software, or as a mixture of both.

With the growing complexity of applications, it was soon discovered that techniques for designing and processing software were quite different from techniques for processing data, and that techniques for acquiring and managing knowledge were different than techniques for data processing and information processing. This discovery led to the development of structured programming, database processing [6], and artificial intelligence [3], [4] in the 1960's and 1970's. Early data and knowledge processing systems were small in scale: modeling, design,

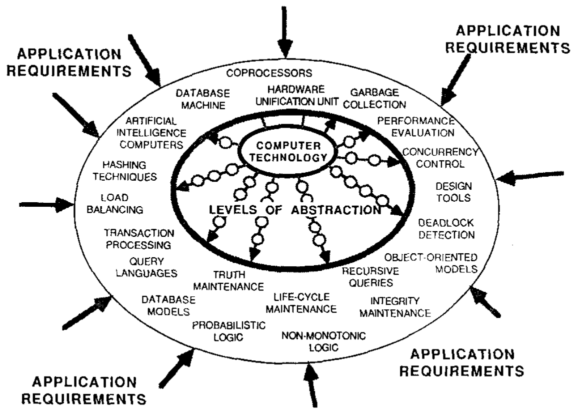


Fig. 1. Problems of various levels of abstraction in knowledge and data engineering.

and management could be handled by one or a few experts.

The ever increasing complexity of applications and information processing systems in the 1980's and the increasing need to capture and manage abstract knowledge require the collective effort of a large number of experts. Data and knowledge can no longer be handled by individuals alone, and its management must be treated as an engineering discipline.

Data and knowledge engineering systems are feasible because of recent advances in technology and computer architecture. Existing technologies, such as VLSI, VHSIC, and fiber optics; and emerging technologies, such as lightwave technologies, sea of gates, three-dimensional VLSI, and superconductivity, promise faster computers, more memory, and higher networking bandwidth. Research on multiprocessing and distributed processing also allows faster and parallel computers to be utilized efficiently.

Studies on knowledge and data engineering, therefore, span a broad spectrum of areas and encompass data, information, knowledge, technology, and products. The subject area is truly dynamic and its emphasis may change as new applications evolve and new technologies become mature.

In this paper we provide an overview of the current research and development directions in knowledge and data engineering. We classify research problems and approaches in this area, and discuss future trends. We do not attempt to survey all related work and references in the area, as the scope of work in this area is extremely broad. Further, since knowledge and data engineering have been applied in many areas, it is not possible to provide a complete discussion of each application.

## II. Characterization of Knowledge/Data Engineering Problems

Solutions to problems in knowledge and data engineering applications depend on the characteristics of these problems. In this section, problems are classified according to three attributes: 1) completeness of knowledge or data in the environment; 2) accuracy of knowledge or data available in the environment; and 3) knowledge about the objective and/or specifications of the problem. Table I shows the eight possible combinations of these three attributes and offers examples for each class. Possible techniques in solving problems in each class are also indicated.

Knowledge/data available in the environment can be complete or incomplete. When it is complete, no additional knowledge is needed to solve the problem. In contrast, when it is incomplete, heuristics must first be applied to find a complete set of knowledge/data. In some cases, the amount of knowledge may be very large or unbounded, and heuristics must be applied to define a restricted set of knowledge/data that can be used in solving the problem. For example, in proving a theorem, it may not be possible to define all of the necessary axioms. As a result, the original problem has to be heuristically modified to prove the theorem based on the *available* axioms.

The knowledge/data available in the environment may be exact or inexact. When it is exact, it can be represented in numerical or logical form. Data/knowledge is inexact when the number of possible cases is infinite, and it is impossible to enumerate or represent all of them. In such cases, before the problem can be solved, heuristics must first be applied to either define a finite number of possibilities or redefine the meaning of exactness so that what is available can be treated as exact.

For example, the birthdays of people in a group are points in a time spectrum, which must first be converted into real numbers of finite precision before the youngest person can be found. In a second example, recognizing an object in a given image involves incomplete data because there are an infinite number of possible orientations and features of the object concerned. To solve the problem, it is necessary to identify a finite and reasonable number of features of the object and heuristically match them with those found in the image. As another example, finding points with the lowest energy in a bounded amount of space is a problem with incomplete and inexact data. Although there is a finite number of particles in such a space, it is not possible to examine the energy value of each. Measurements of a finite degree of precision will be made for a reasonable number of points, and interpolations will be made for intermediate points. Heuristics are, therefore, applied to redefine the meaning of exactness and completeness in the original problem.

After knowledge/data used in solving the problem is defined or restricted, the problem can be solved by finding a solution in the given solution space. This can be represented as the objective of what the solutions must achieve and the specifications that the solutions must satisfy.

An objective of a problem may be well-defined or ill-defined. A well-defined objective can be represented exactly in terms of the measurable parameters of the problem environment so that it is possible to compare the quality of one solution to another. An ill-defined objective may

TABLE I
CHARACTERIZATION OF KNOWLEDGE AND DATA ENGINEERING PROBLEMS

| COMPLETENESS | EXACTNESS | OBJECTIVE/ SPECIFICATIONS | EXAMPLES | PROCESSING TECHNIQUES |
|---|---|---|---|---|
| Complete | Exact | *Well-defined* | *Querying a relational database* | Parallel processing, combinatorial searches |
| | | Ill-defined | Querying a statistical database | Heuristics must first be applied to define objective/specifications. Learning algorithm is one possible heuristic. |
| | Inexact | Well-defined | Finding the youngest person in a group; Recognizing an object in a given image | Heuristics must first be applied to find better data/knowledge or restrict data to a fixed degree of precision. |
| | | Ill-defined | Recognizing language in a voice pattern | See comment above for ill-defined objective/specifications and inexact data. |
| Incomplete or Unbounded | Exact | Well-defined | Proving a theorem | Heuristics must first be applied to find a complete set of data/knowledge. If this is not possible, heuristics must be used to define a restricted set of knowledge/data that can be used in solving the problem. |
| | | Ill-defined | Deciding where to process a task to minimize the average response time | See comments above for incomplete data and *ill-defined* objective/specifications. |
| | Inexact | Well-defined | *Finding points with* the lowest energy in a bounded space | *See comments above for* incomplete and inexact data. |
| | | Ill-defined | Predicting changes in weather; Reasoning with fuzzy logic | See comments above for incomplete and inexact data and ill-defined objective/specifications. |

involve either parameters that cannot be expressed in measurable terms or an unknown relationship among measurable parameters. As a result of these conditions, it is not possible to compare the quality of alternative solutions. An ill-defined objective must first be heuristically transformed into a well-defined one before the problem can be solved. This may involve restricting consideration to measurable parameters and defining the objective as a function of these parameters.

The specifications that the solution must satisfy may also be either well-defined or ill-defined. In a well-defined set of specifications, it is possible to enumerate all candidate solutions. In an ill-defined set of specifications, a potentially infinite solution space is defined. Problems with ill-defined specifications must first be heuristically transformed so that all specifications are defined precisely in terms of measurable parameters of the environment.

The querying of a statistical database is an example of a problem with an ill-defined objective because one can only obtain probabilistic rather than precise answers to queries. The recognition of natural language from a voice pattern is a problem with inexact data and an ill-defined objective because the inputs are represented as analog signals that are prone to errors, and because there are an infinite number of possible pronunciations of a word. In these conditions, the specifications of the solution space cannot be defined precisely. For this problem, only a finite number of possible pronunciations can be tested. The decision of where a task should be performed in a distributed computer system is another problem with an ill-defined objective because the relationship between the objective (minimizing the response time) and the measurable parameters (workload statistics) is unknown. A heuristic function relating the response time and the measurable quantities must first be defined before the problem can be solved. Finally, the prediction of weather is a problem which may involve an infinite amount of inexact input data and ill-defined objectives and specifications.

## III. RESEARCH ON KNOWLEDGE AND DATA ENGINEERING

The goal of research in knowledge and data engineering is to study and design systems to support knowledge and data engineering applications. Approaches to achieving this goal range from theoretical analysis, mathematical modeling, simulations, and prototyping, to complete system integration.

The design process of a knowledge and data engineering system is extremely complex and iterative. It is not possible to classify all of the variations of this process without oversimplifying it. However, the resulting design has notable characteristics depending on its starting point, which usually dictates the nature of the final result and can be used as a classification scheme for results in this area. The starting point can be either application-driven or technology-driven.

In an application-driven or a top-down approach, research is focused on first identifying and refining the requirements of the problem. The knowledge necessary for solving the problem is then acquired, and algorithms are designed and mapped to physical systems. In this approach, the capabilities of the resulting system are a good match for the requirements of the application. However, in some instances the requirements are specified in such a way that a system cannot be physically realized. In this case the design process has to be iterated through repeated refinement of the requirements and designs.

In contrast, a technology-driven or a bottom-up approach determines technological capabilities and limitations first. These limitations lead to the design of realizable systems on which the applications can be mapped. Tradeoffs on performance are then evaluated, and one of the candidates is selected as the target system. The problem with this approach is that the resulting design may not be a perfect match for the application requirements. The design process has to be iterated by modifying the design until the application requirements are satisfied.

Research on data and knowledge engineering addresses the theory, analysis, design, development, evaluation, and maintenance of new data and knowledge management techniques, methodologies, and systems that can support specialized applications in a cost-effective manner. It also emphasizes a better understanding of problem solving methods with respect to the applications concerned. Related issues in accomplishing these goals include studies on the theoretical aspects, design tools and methodologies, design tradeoffs, representation and programmability, algorithms and control, reliability and fault tolerance, and designs using existing and emerging technologies. In the following text, we discuss each of these issues by stating its goals and showing some representative examples. It must be pointed out that all of these issues must be considered in designing a complete working system.

Fig. 2 depicts a hierarchical organization of studies in knowledge and data engineering. The problem to be solved may be specified in an imprecise form and must be successively refined into a form that can be implemented physically. Applicable methods for such refinement include synthesis, analysis, optimization, verification, validation, testing, and maintenance. Representation on a higher level is more influenced by the applications, while representations on a lower level are influenced by technological constraints.

*Theory:* Theoretical studies involve abstracting the problem domain and studying the properties and limitations of the abstract problem. Such an abstraction is often necessary to transform the original problem, which is too complex to be studied in its entirety, into a simpler and more manageable form. Moreover, since the abstract model may cover a number of different problems in the class, the results developed will be applicable to these problems as well. Theoretical results can be used to guide the selection of algorithms and design tradeoffs.

*Programmability and Representation:* This issue involves choosing or designing the appropriate knowledge and data representation schemes and developing the necessary hardware and software support mechanisms. A choice must be made between using an existing representation scheme, such as one that is declarative, procedural, functional, or object-oriented; or designing a new representation scheme. The scheme may have to take into account uncertain, fuzzy, or incomplete knowledge and data in the application, and may need to support data and knowledge engineering applications in a sequential or parallel environment.

Many of the issues of representation should be considered early in the design process when the problem semantics are still available. An essential issue to consider is specifying a minimal amount of information to allow control to be carried out efficiently in an implementation. This knowledge should preferably be specified in a form that is independent of the computer architecture on which the knowledge and data management system is implemented. Detection of this information for efficient control may have to be delayed until run time when more accurate knowledge about the resources is available.

*Design Tools and Methodologies:* The design of a knowledge-based and database system is a complex task, which can be greatly simplified by using computer-aided design tools. Systematic methods must be used for analyzing requirements, representing specifications, partitioning problems into a manageable size, designing algorithms, mapping algorithms into physical data and knowledge management subsystems, and maintenance of these systems. The process is iterative and stops when problem requirements, such as flexibility, extendability, reconfigurability, user friendliness, reliability, and evolvability; and technological limitations, such as chip size, bandwidth, and packaging, are met.

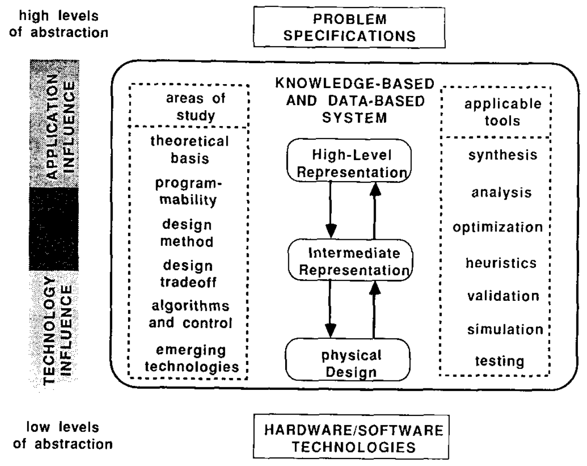


Fig. 2. Design of knowledge-based and data-based systems.

*Design Tradeoffs:* Design tradeoffs for data and knowledge management systems involve tradeoffs on hardware and software implementations, representation schemes, use of software languages, computational power of processors, communication bandwidth, storage capacity, cost, design of specialized functional units, operating system environment, and use of hardware technologies. A judicious selection of the components to be used in the system must be made in order to obtain a high performance system with acceptable cost.

*Algorithms and Control:* This issue involves the design and evaluation of algorithms and control strategies for efficient and reliable operations in data and knowledge management systems. It also involves methods to acquire, test, store, access, and maintain reliable data and knowledge for a given application.

The design process is usually hampered by imprecise or incomplete knowledge at design time, which necessitates developing and using heuristics for control. These heuristics may underutilize resources and cause anomalous behavior with respect to resources. In the latter case, increasing the resources available to a system may result in worse performance by the heuristics (with respect to the objective).

Another problem in the design of algorithms is that they depend largely on the characteristics of the application involved and the experience of the designers. Automated methods for designing and evaluating algorithms are not available at this time. Computer-aided tools, such as automated learning methods, may be very useful in adapting existing algorithms to new applications.

*Emerging Technologies:* This issue involves exploration of the limitations and impact of emerging technologies on the design, evaluation, programmability, and application span of data and knowledge engineering systems. Emerging technologies, such as optical processing [9], artificial neural networks [2], high bandwidth optical links,

TABLE II
EXAMPLES OF DATA AND KNOWLEDGE ENGINEERING ISSUES

| |
|---|
| **Theoretical Basis** |
| Recursive queries |
| Database models |
| Logic databases and problem solving |
| Non-monotonic database and knowledge base |
| **Programmability and Representation** |
| Truth maintenance |
| Object-oriented models |
| Query, design, and visualization languages |
| Knowledge and data representation schemes |
| **Design Methods and Tools** |
| Design methodology |
| Life-cycle maintenance |
| Computer-aided design tools |
| Knowledge acquisition schemes |
| System integration and modeling |
| Statistical databases and knowledge bases |
| **Design Tradeoffs** |
| Standardization |
| Application experience |
| Real-time knowledge bases and databases |
| Performance evaluation techniques and tools |
| Integrated voice, image, and data processing |
| Intensional versus extensional data processing |
| **Algorithms and Control** |
| Hashing techniques |
| Integrity maintenance |
| Transaction processing |
| Data communication aspects |
| Machine learning algorithms |
| Distributed database control |
| Parallel and distributed processing |
| Garbage collection and virtual memory management |
| Fault detection, isolation, checkpointing and recovery |
| **Emerging Technologies** |
| Database machines |
| Dictionary machines |
| Artificial intelligence computers |
| Database and knowledge-base coprocessor |

new packaging technologies [5], large memory devices, and new architectural concepts, may impact the ways that design tradeoffs are performed and algorithms are developed. They may also result in different design methods, problem solving strategies, and control algorithms for future data and knowledge engineering systems and impact the evolution of current systems.

Table II shows a few examples of some of the issues that are studied in knowledge and data engineering and are intended to be covered by the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. Some of these issues are more exploratory in nature, while others may be more related to industrial practices.

## IV. FUTURE CHALLENGES

Improvements in knowledge and data engineering systems can come from faster technologies, better representation schemes, more efficient algorithms, automated software design methods, and better hardware/software architectures integrated with the available technologies.

*Emerging Technologies:* The basis for any computer system is the technology in which it is implemented. The design of a system is often driven by its cost; hence, the fastest technologies, subject to cost specifications, are used. New technologies may give higher performance but are often prohibitively expensive. These improvements will likely bring a two to three orders of magnitude improvement in computational speeds in the next decade.

*Knowledge Representations:* Many new representation schemes have evolved in the recent past. These schemes may feature tools for knowledge and data capture and management. However, they are usually not directed towards any specific applications and may have to be modified or extended to tailor to the applications and computational environment. Another major problem is the lack of an overall technique to guide the evaluation and selection of a representation scheme. Research in this area could prove extremely valuable. Learning techniques for incorporating new knowledge about application domains into current solutions in a knowledge-intensive application may also impact knowledge and data engineering.

*Algorithms:* Research in the area of application-specific algorithms will have the greatest potential for speeding the solution of the given application. The development of new and improved algorithms for an application can be seen as finding alternative ways to incorporate knowledge about the application domain and the technology into the computer solution.

The design of better algorithms and knowledge representation schemes is an important complement to the tremendous potential offered by emerging technologies. Extension in the limit of processing power by new technologies is valuable, especially for real-time systems. However, the two to three orders of magnitude speed improvement offered by these technologies in the next decade will not greatly impact the size or type of knowledge and data engineering applications that are addressed today. Many of these applications involve huge search spaces of an exponential size; two to three orders of magnitude increase in computational speed will do little to extend the size of a solvable instance of such a problem [7].

Table III shows, for a given problem, the extended problem size that can be solved by faster or parallel computers in the same amount of time as a single computer solving the same problem and assuming that linear speedup is possible. It illustrates that, for more complex algorithms, faster computers and parallel processing alone are only useful to improve the turn-around time of algorithms that can already be evaluated on existing computer systems.

*Software Architecture:* Software architecture is an integral part of a knowledge and data management system. Generation of new software environments, tools, and languages will probably rely on amalgamation of known representation techniques and software design methodologies. Software development systems and automated intelligent assistants represent prime areas for advancement of data and knowledge engineering applications. The problems of verification and validation and continuous maintenance of programs, knowledge, and data are important related topics.

TABLE III
PROBLEM SIZE SOLVABLE BY SEQUENTIAL AND PARALLEL PROCESSORS ASSUMING LINEAR SPEEDUP AND THE SAME AMOUNT OF TIME AS SEQUENTIAL PROCESSING

| **Algorithm Complexity** | **Number of Processors** | | |
|---|---|---|---|
| | 1 | 2 | N |
| $N$ | $N$ | $2N$ | $N^2$ |
| $N^2$ | $N$ | $\sqrt{2}N$ | $N^{1.5}$ |
| $N^k$ | $N$ | $2^{1/k}N$ | $N^{1+1/k}$ |
| $2^N$ | $N$ | $N+1$ | $N+\log_2 N$ |

*Hardware Architectures:* As with software, hardware architectures are often based on known design techniques such as parallel processing and pipelining. Innovation for new architectural concepts may be made possible by new and emerging technologies, which affect the cost effectiveness of previous designs.

New hardware architectures are best utilized for operations that the computer performs frequently. Counter to intuition, identification of these tasks is very difficult. Operations may be instructions, parts of instructions, groups of instructions, or frequently recurring tasks. Identification of new and valid areas for development of new hardware architectures is an important area of research.

*System Design:* System-level design is often based on an overall design philosophy and may contain, for example, a mix of different computational models, and general-purpose and special-purpose functional units. An important driving force is the cost-effectiveness of the design and its technological feasibility. A major difficulty, however, lies in integrating designs with different knowledge representation schemes, software architectures, and hardware components, and evolving systems as technologies change.

## V. AN INVITATION TO PARTICIPATE

We sincerely and earnestly invite you to actively participate in this TRANSACTIONS. We need authors to submit high-quality papers; without them, this TRANSACTIONS will not survive. We need reviewers to help authors refine their work and to point out improvements. Their generous devotion of time is highly commended. We need guest editors to focus special issues in the forefront of science, technology, and engineering. Their participation is critical in this ever-changing area of knowledge and data engineering. We need editors to devote their time to processing papers and to recommend the best articles to publish in our TRANSACTIONS. We need researchers and practitioners to serve on the Advisory Board, who will advise us on policies and new research directions. They also represent a diversified source of expertise that will guide our TRANSACTIONS into the 1990's. Finally, and most importantly, we need readers and subscribers, like you, to help us maintain this TRANSACTIONS in a sound financial condition. Your support, help, and comments will be most crucial in keeping the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING a forefront in this area.

We, now, take the privilege of introducing our distinguished members of the Advisory Board. A list of the Advisory Board can be found on the inside front cover of this issue. The objective of the Advisory Board is to provide us with timely guidance, orientation, and focus. The members of the Board characterize the interdisciplinary nature of our TRANSACTIONS. This is vital as our future projects, particularly, the big ones, will require a multidisciplinary team of experts to design, develop, and implement. We do not need to say anything more about our Advisory Board members, since you know them all by their outstanding contributions.

We are fortunate to have a very distinguished group on our Editorial Board. A list of the editors of this TRANSACTIONS and their short biographies is included at the end of this issue.

We welcome papers that present original results in research and development and survey the state of the art in areas relevant to knowledge and data engineering. Papers submitted may be interdisciplinary, application-oriented, and technology-related. We strongly advise that results presented should include some practical ramifications, experience, and impact on the engineering discipline. Papers that present purely theoretical results with no practical development will be of interest to only a very small audience and, hence, will not be a major focus of this TRANSACTIONS.

Papers that have not previously been published in another journal, as well as papers that have been published in Conference Proceedings, Digests, and Records and that have undergone substantial revision may be submitted for consideration. Invited papers from leading authorities in the knowledge and data engineering area will also be published.

Three types of papers will be published:

1) regular technical papers (25–35 double-spaced pages, including figures, tables, and references): a) papers with extensive original results; and b) in-depth surveys, which contribute to the understanding and advances in the knowledge and data engineering area;

2) concise short papers (maximum 12 double-spaced pages): papers with results that are important and original and are presented in a concise form;

3) correspondence items (maximum 3 double-spaced pages): comments on previously published papers, short extensions to current results, critiques on previous results, responses from authors, and corrections to previously published papers.

Further details on submitting papers and proposals for special issues can be found on the inside back cover of this premier issue.

## VI. ABOUT THIS ISSUE

This issue is made up of ten invited papers from leading researchers in the knowledge and data engineering area.

The topics covered range from historical perspectives to seminal surveys to future trends. They represent ideas and visions in the forefront of knowledge and data engineering.

The paper by Charles W. Bachman, ''A personal chronicle: Creating better information systems, with some guiding principles'' shows a fifty-year passage, describing some of the challenges and identifying some of the guiding principles discovered along the way. It shows some of the valuable life-long experience and perspectives the author has in designing data engineering systems.

The paper by Michael Stonebraker, ''Future trends in database systems'' presents the likely evolution of commercial data managers over the next several years. Questions answered include: 1) why SQL has become an intergalactic standard; 2) who will benefit from SQL standardization; 3) why the current SQL standard has no chance of lasting; 4) why all database systems will be distributed soon; 5) why new technologies are likely to be commercialized; and 6) why vendor independence may be achievable.

The paper by David K. Hsiao and Magdi N. Kamel, ''Heterogeneous databases: Proliferations, issues, and solutions'' characterizes heterogeneous databases as the inevitable consequence in replacing the traditional processing practice with the modern database management. The authors provide a taxonomy of solutions to the problems and issues of heterogeneous databases and discuss some of the current research directions in the area.

The paper by Edmund H. Durfee, Victor R. Lesser, and Daniel D. Corkill, ''Trends in cooperative distributed problem solving'' examines this emerging area and surveys the important approaches and empirical investigations in this area. Areas covered include negotiation, functionally-accurate cooperation, organizational structuring, multiagent planning, sophisticated local control, and theoretical frameworks.

The paper by Douglas B. Lenat, ''Ontological versus knowledge engineering'' centers around the difference in the task of building a huge knowledge base from that of building a number of small knowledge bases. The discussion opens the realm of ontological engineering, which leads to the conclusion that a tool-box approach may be a viable approach to the problem.

The paper by Lotfi A. Zadeh, ''Knowledge representation in fuzzy logic'' examines the shortcomings of bivalent logic and presents fuzzy logic as an extension of classical logical systems which provides an effective conceptual framework for dealing with the problem of knowledge representation in an environment of uncertainty and imprecision. It provides a summary of the basic concepts and techniques underlying the application of fuzzy logic to knowledge representation and describes a number of examples relating to its use as a computational system for dealing with uncertainty and imprecision in the context of knowledge, meaning, and inference.

The paper by Frederick Hayes-Roth, ''Towards benchmarks for knowledge systems and their implications for data engineering'' suggests criteria for good measures for extrinsic concerns, which include advice quality, reasoning correctness, robustness, and solution efficiency, and intrinsic concerns, which center on elegance of knowledge base design, modularity, and architecture. Benchmarks and ways to achieve these measures through the design of knowledge system experiments are proposed.

The paper by P. Bruce Berra, Arif Ghafoor, Pericles A. Mitkas, Slawomir J. Marcinkowski, and Mohsen Guizani, ''The impact of optics on data and knowledge base systems'' addresses the impact of the emerging optical technologies on data and knowledge base systems. Issues and solutions related to storage, interconnection, and processing are presented.

The paper by Richard J. Fateman, ''A review of Macsyma'' reviews the successes and failures of the Macsyma algebraic manipulation system from the point of view of one of the original contributors. Topics covered include input/output, language semantics, knowledge-adjunction, mathematical semantics, the model of the user, and possible future directions for algebraic manipulation system building.

Lastly, the paper by Stefano Ceri, Georg Gottlob, and Letizia Tanca, ''What you always wanted to know about Datalog (and never dared to ask)'' surveys the research on Datalog and indicates what is still needed to apply Datalog to real-life problems. Various optimization methods for efficiently evaluating Datalog queries are also discussed.

## References

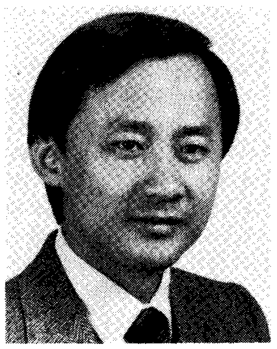
[1] ''Special Issue on Data Engineering,'' *IEEE Computer Mag.*, vol. 19, no. 1, Jan. 1986.
[2] Defense Advanced Research Project Agency, DARPA Neural Network Study, Lincoln Lab., Mass. Inst. Technol., Lexington, MA, July 1988.
[3] A. Barr and E. A. Feigenbaum, *The Handbook of Artificial Intelligence*, Vols. 1, 2, and 3. Los Altos, CA: Kaufmann, 1981, 1982.
[4] J. McCarthy, ''Program with common sense,'' in *Mechanization of Thought Processes.* London: Her Majesty's Stationery Office, 1959, pp. 75-84.
[5] J. F. McDonald, H. J. Greub, R. H. Steinvorth, B. J. Donlan, and A. S. Bergendahl, ''Wafer scale interconnections for GaAs packaging—Applications to RISC architecture,'' *IEEE Computer Mag.*, vol. 20, no. 4, pp. 21-35, Apr. 1987.
[6] J. D. Ullman, *Principles of Database Systems.* New York: Computer Science Press, 1984, pp. 211-267.
[7] B. W. Wah, G. J. Li, and C. F. Yu, ''Multiprocessing of combinatorial search problems,'' *IEEE Computer Mag.*, vol. 18, no. 6, pp. 93-108, June 1985.
[8] H. Webster, *Webster's II New Riverside University Dictionary.* Boston, MA: Riverside, 1984.
[9] L. C. West, ''Picosecond integrated optical logic,'' *IEEE Computer Mag.*, vol. 20, pp. 34-47, Dec. 1987.
[10] G. Wiederhold, ''Knowledge and database management,'' *IEEE Software Mag.*, vol. 1, no. 1, pp. 63-73, Jan. 1984.

C. V. RAMAMOORTHY,
*Editor-in-Chief*
BENJAMIN W. WAH,
*Associate Editor-in-Chief*

**C. V. Ramamoorthy** (M'57–SM'76–F'78) received the M.S. and Ph.D. degrees in applied math from Harvard University, Cambridge, MA, and the M.S. degree in mechanical engineering from the University of California, Berkeley.

Since 1972 he has been a Professor in the Department of Electrical Engineering and Computer Science at the University of California, Berkeley. Previously, from 1967 to 1972, he was a Professor of Electrical Engineering and Computer Science at the University of Texas, Austin. From 1956 to 1971 he was associated with Honeywell's Computer Division (now Honeywell Bull), Waltham, MA, where he last held the position of Senior Staff Scientist. He also held the Grace Hopper Chair and C.D.C. Distinguished Visiting Professorships at the Navy Post Graduate School and the University of Minnesota, respectively.

Dr. Ramamoorthy served as First Vice President of the IEEE Computer Society, Editor-in-Chief of the IEEE TRANSACTIONS ON SOFTWARE ENGINEERING from 1983 to 1987, and is currently Editor-in-Chief of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. He is presently a participant in the Distributed Intelligent Networking Group of the U.S. Army Strategic Defense Command.



**Benjamin W. Wah** (S'74–M'79–SM'85) received the Ph.D. degree in computer science from the University of California, Berkeley, in 1979.

He has been on the faculty of the School of Electrical Engineering at Purdue University, West Lafayette, IN, between 1979 and 1985. He is currently a Professor in the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory of the University of Illinois at Urbana-Champaign. Between 1988 and 1989, he served as a Program Director of the Microelectronic Systems Architecture Program, National Science Foundation. He has published extensively in the areas of computer architecture, parallel processing, artificial intelligence, distributed databases, and computer networks.

Dr. Wah is Associate Editor-in-Chief of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, an Area Editor of the *Journal of Parallel and Distributed Computing*, and an Editor of *Information Sciences*. He serves as a member of the Governing Board of the IEEE Computer Society and a program evaluator for ABET (computer engineering) and CSAC (computer science). Previously, he served as an Editor of the IEEE TRANSACTIONS ON SOFTWARE ENGINEERING and a Distinguished Visitor of the IEEE Computer Society.