AI Magazine Volume 11 Number 1 (1990) (© AAAI)

Directions in AI Research and Applications at Siemens Corporate Research and Development

Wolfram Buettner, Klaus Estenfeld, Hans Haugeneder, and Peter Struss

■ *Many barriers exist today that prevent effective industrial exploitation of current and future AI research. These barriers can only be removed by people who are working at the scientific forefront in AI and know potential industrial needs.*

The Knowledge Processing Laboratory's research and development concentrates in the following areas: (1) natural language interfaces to knowledge-based systems and databases; (2) theoretical and experimental work on qualitative modeling and nonmonotonic reasoning for future knowledge-based systems; (3) application-specific language design, in particular, Prolog extensions; and (4) design and analysis of neural networks.

This article gives the reader an overview of the main topics currently being pursued in each of these areas.

Many barriers exist today that prevent effective industrial exploitation of current and future AI research. These barriers can only be removed by people who are working at the scientific forefront in AI and know potential industrial needs.

The Knowledge Processing Laboratory within Siemens Corporate Research and Development in Munich, Germany, aims at breaking down these barriers. With over 350,000 employees and a product line ranging from coffee makers to mainframe computers, Siemens is one of the world's leading high-tech companies. Its interest in leading-edge technologies such as AI is understandable.

The Knowledge Processing Laboratory's research and development concentrates in the following areas: (1) natural language interfaces to knowledge-based systems and databases; (2) theoretical and experimental work on qualitative modeling and nonmonotonic reasoning for future knowledge-based systems; (3) application-specific language design, in particular, Prolog extensions; and (4) design and analysis of neural networks.

The lab's 26 researchers are organized into four groups corresponding to these areas. Together, they provide Siemens with innovative software technologies, appropriate applications, prototypical implementations of AI systems, and evaluations of new techniques and trends.

In addition to its responsibilities to the company, the lab aims at establishing close contacts with the outside scientific community. In part, this contact is made by sharing technical results and advances in AI technology and its theoretical foundations. Furthermore, lab researchers support a two-way exchange of information through publications and talks on current work as well as activities as referees, teachers at tutorials, and conference organizers.

The remainder of this article gives the reader an overview of the main topics currently being pursued in each of the four groups.

Natural Language Technology

Current state-of-the-art natural language interfaces have many limitations. Among these are a linguistic coverage that is too narrow, restricted types of dialogue interaction, insufficient robustness, and modeling of the application domain that is too implicit.

To develop interfaces that do not exhibit these limitations, the Natural Language Technology Group is engaged in basic research and in the development of actual applications. The theoretical work encompasses such elements as the development of expressive formal representations for linguistic knowledge and empirically tested, formally explicit descriptions of linguistic phenomena. The computational work concerns questions of adequate processing models and algorithms, as embodied in the actual interfaces being developed.

These topics are explored in the framework of three projects: The natural language consulting dialogue project (Wisber) takes up research-oriented topics (descriptive grammar formalisms and linguistically adequate grammar specification, handling of discourse, and so on), the data-access project (Sepp) focuses on the practical application of state-of-the-art technology, and the work on grammar-development environments (Ape) centers on the problem of adequate tools for specifying linguistic knowledge sources and efficient processing methods.

Wisber is jointly funded by the German government and several industrial and academic partners. The goal of the project is to develop a knowledge-based advice-giving system in the domain of financial consultation.

> *The group's general strategy for validating its theoretical work is to build prototypical tools and integrate them into a system.*

The natural language analysis component is the laboratory's responsibility. It is based on a lexical-functional grammar (LFG)–type parser (Block and Hunze 1986; Block and Haugeneder 1988) but handles long-distance phenomena more in the spirit of the slash categories of the generalized phrase structure grammar. It uses discourse representation theory (DRT) as the framework for its semantic and discourse representation (Hunze 1988; Frederking and Gehrke 1988). A KL-TWO–type representation formalism is used for conceptual modeling. A grammar compiler developed by the laboratory allows grammars to be perspicuously written but efficiently processed. Because morphology is of critical importance in German, we also developed a morphological analyzer using the same formalism as for syntactic analysis. Additionally, we implemented a component for handling and accessing large lexicons on the basis of discrimination trees and random-access methods (Gehrke and Block 1986). A sizable, restrictively formulated German grammar is being developed as part of this work using government and binding concepts within an LFG representation (Schachtl 1988). These components run in Interlisp-D on a Siemens 58xx (Xerox 1100) workstation.

Sepp (SESAM preprocessor) is a natural language interface to database systems supporting the standard query language SQL (Block and Schlereth 1987). Included are tools for adjusting the domain-independent language processor to a specific database. Sepp functions as a translator, producing SQL queries in a two-phase process. Natural language queries are mapped into SQL queries through an intermediate abstract, object-centered internal representation.

The current version of Sepp is being used in a pilot project by an organization outside Siemens to access information in its large, real-world database. This experience will give us feedback on the system's language coverage, habitability, and overall facilities. The next version of this interface will employ left-corner parsing, making use of bottom-up information in addition to the top-down information already used. Work also continues on extending Sepp's linguistic coverage, especially with regard to quantifiers, ellipses, and comparative constructions (the latter are currently handled by semiformal expressive features). Enlarging the functionality and user friendliness of Sepp's domain-tailoring tools will also be tackled (Schmid 1988). The development of the system is being performed in Prolog on SUN workstations, with Siemens MX (Unix-based) computers as target machines.

The Ape (augmented transition network [ATN] programming environment) grammar-development environment is a highly interactive development and testing environment for the ATN formalism based on an active chart parser (Haugeneder and Gehrke 1986). This system allows the specification and debugging of network-style grammars using an extensive graphic interface and provides a highly flexible tool for defining and testing various control strategies (Haugeneder and Gehrke 1987) using an agenda control mechanism. These capabilities have been used extensively in the development of heuristic search strategies that are independent of the underlying grammar formalism (Haugeneder and Gehrke 1988). Ape is implemented in Interlisp-D on a Siemens 58xx.

To provide a focus for future work, we have adopted the ambitious goal of developing a linguistic core processor (LCP) that will bring together the results of the work in the three preceding projects. LCP will be an integrated, application-neutral natural language–processing component having linguistic capabilities adequate for a variety of interface uses. LCP is envisioned not only as being parameterized with respect to domain but also as being usable as a natural language interaction component in multimodal user interfaces. The natural language facilities of LCP will include the analysis of natural language input as well as the generation of natural language output. In its final version, it will have comprehensive coverage and be robust in the face of ungrammatical input, attentive to its discourse context, and alert to the implications of a variety of speech acts. Of course, the initial version will be significantly limited compared to the eventual LCP, with development proceeding incrementally.

Participants: H. Haugeneder, H. U. Block, R. Frederking (until June 1989), M. Gehrke, R. Hunze, S. Schachtl, and L. Schmid.

Advanced Reasoning Methods

The Advanced Reasoning Methods Group is engaged in the areas of qualitative reasoning and truth maintenance. Both are key issues in building systems that use deep knowledge as opposed to the conventional rules of thumb extracted from domain experts. The group participates in the TEX-B project, Foundations for Expert Systems in Technical Domains, which is funded by the German government.

The results obtained to date in the area of qualitative reasoning systems include a framework for component-oriented device modeling with a means for structuring models in a hierarchical and view-oriented fashion and focusing the analysis of these models (Struss 1987). A systematic analysis of contemporary qualitative reasoning methods within a general mathematical model of qualitative calculi (based on algebraic and differential equations) uncovered some inherent limitations to these approaches (Struss 1988c, 1988d, 1988e). Continued joint work in this area with researchers from XEROX Palo Alto Research Center led to the thesis that some of these limitations can be overcome by radically departing from the concepts of differential calculus and focusing on commonsense physics (Huberman and Struss 1989).

In the area of truth maintenance, the group's main contributions are extensions to justification-based truth maintenance systems (JTMSs) and assumption-based truth maintenance systems (ATMSs). These efforts include integrating nonmonotonic justifications into the essentially monotonic ATMS as well as encoding disjunctive and negative information (Dressler 1988a, 1988b). Justification schemata (Freitag and Reinfrank 1988; Reinfrank and Freitag 1988) add limited first-order capabilities to the inherently propositional TMSs and constitute an alternative choice for implementing systems based on ATMSs or JTMSs. Controlling the inferences of TMS-based problem solvers is a central problem of the field. The control strategy for ATMS, which is based on local control guards and a global focus (Dressler and Farquhar 1989), enables the implementation of tight problem solver control. It is general enough to represent all control regimes that have been proposed for ATMS in the literature.

The statement of a problem should at the same time be an executable computer program that solves the problem.

Based on these TMSs, the group has developed constraint systems, in particular, for the propagation of temporally indexed values (Decker 1988; Dressler and Freitag 1989).

A recent and noteworthy theoretical contribution of the group is a logical theory of truth maintenance (Reinfrank and Dressler 1989). By translating JTMSs, as well as ATMSs, into a variant of nonmonotonic logic, the theory puts truth maintenance on a firmer ground than was previously possible.

The group's general strategy for validating its theoretical work is to build prototypical tools and integrate them into a system. This practical system functions as a workbench, providing a basis for experimentation and empirical feedback into further theoretical investigations.

The practical application of the group's techniques in this case is model-based diagnosis (Struss 1988a, 1988b, 1989a, 1989b). Currently, the diagnostic system under development is mainly evaluated in the domain of thyristor circuits for the power supply and control of direct-current motors. This example confronts approaches to model-based diagnosis with several important challenges, such as feedback, the changing of device topology, temporal reasoning, imprecise observations, and multiple models. The approach is based on an extended version of de Kleer and Williams's general diagnostic engine. As implemented, this framework deals with hierarchical models and allows focusing within these models (Struss 1988a, 1988b), handles dynamic device models (Dressler and Freitag 1989), and integrates fault models exploiting the extended ATMS (Struss and Dressler 1989).

Future work involves investigating and exploiting the relationships between qualitative reasoning and nonmonotonic reasoning, for example, a description of process models using situation-action structures. The work will emphasize the dynamic aspects of the application domains, for example, by considering actions taken by a diagnostician that shift the device under consideration into different states. Paradigms from both qualitative and nonmonotonic reasoning must be integrated.

Participants: P. Struss, R. Decker (until June 1989), O. Dressler, H. Freitag, M. Reinfrank, and A. Farquhar (guest researcher until July 1989).

Prolog Extensions for Scientific and Technical Applications

The ideal of declarative programming can be summarized as follows: The statement of a problem should at the same time be an executable computer program that solves the problem. This ideal has been approximated by Prolog, but there has been little evidence that Prolog could achieve the ideal in a technical industrial environment, for example, very large-scale integrated circuit design, communications systems, or knowledge-based design of technical systems.

The Language Extension Group has tried to meet the need for declarative programming in these areas by suitably extending Prolog. On the basis of theoretical research (Buettner 1988; Enders 1989), a new system, PROLOG-XT (Estenfeld and Ruckert 1988), was designed and implemented.

Most current commercial Prolog systems lack expressiveness, efficient control, and ease of use. Many complex application domains cannot be adequately modeled. Unification only works on a syntactic level (Herbrand terms); there is no possibility to reflect, for example, the algebraic properties of special application domains. Representing hierarchically structured knowledge for knowledge-based systems is possible only on top of Prolog. Inheritance mechanisms

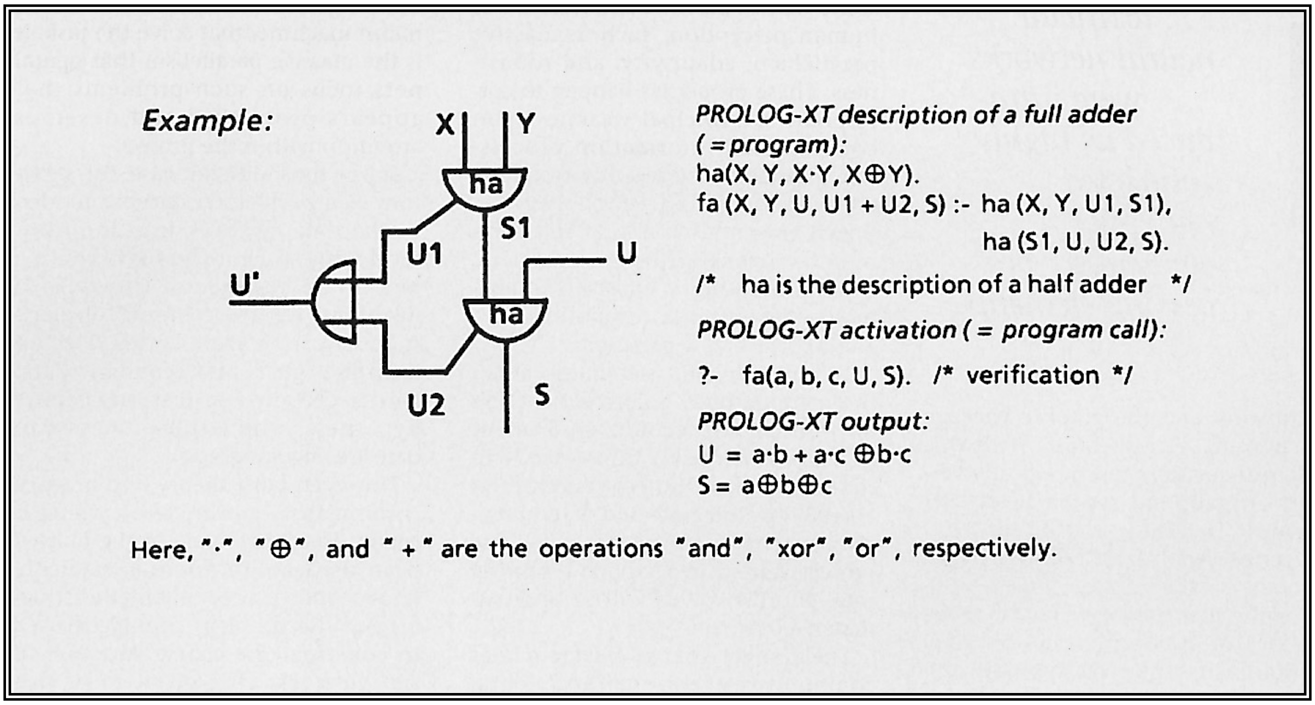

Figure 1. Modeling of Input-Output Relations in a Digital Circuit Using Prolog-XT.

implicitly available in object-oriented programming languages have to be explicitly coded.

As a control mechanism, only the Prolog standard computation rule is available; no clean mechanisms are at hand to avoid superfluous computation steps. Comfortable programming environments and graphic tools for building application-specific user interfaces are the exception. The facilities for closely coupling Prolog programs with more conventional software are usually only rudimentary.

The new Prolog system PROLOG-XT is intended to cope with these deficiencies. The expressiveness of standard Prolog is extended in several ways. Domain-specific semantics can be modeled through algebraic terms, and unification is extended to finite algebraic models. Thus, for example, the input-output relations of digital circuits on the gate level can be modeled by Boolean functions and processed in PROLOG-XT by Boolean unification (figure 1).

Object-oriented programming is integrated into PROLOG-XT through typed variables and typed unification on a compiler level. Hierarchically structured knowledge can thus be represented in a natural way and processed efficiently.

To overcome the standard (static) computation rule and to keep it flexible, coroutining is incorporated into PROLOG-XT on the basis of a delay mechanism. Delay conditions that are checked at run time and might possibly delay the execution of a goal can be added to any user-defined Prolog procedure. In most cases, it is the noninstantiation of goal parameters at run time that is responsible for the delay. The idea is to postpone the execution of such a goal until its parameters are sufficiently instantiated. Unnecessary computation steps can thereby be avoided.

With PROLOG-XT, application-related user interfaces can be easily constructed. A set of integrated built-in predicates provides full access to standard graphics and window management tools. A graphic debugger for the objects allows comfortable program development. External language code can easily be accessed through the C language interface. Backtrackable and even delayable C functions can be written and handled this way.

The object system of PROLOG-XT (Enders 1989) is based on order-sorted logic. *Classes* (the object-oriented analogy to sorts), metaclasses, and objects are provided. Reasoning on the class hierarchy (multiple inheritance) is enabled through sorted variables. Procedures are generalized to generic procedures, such that *methods* (procedure parts belonging to a class) can be independently managed. Method selection and inheritance are automatically regulated by sorted variables in the clause heads. To achieve high performance, abstract machine instructions are added that handle sorted variables and indexing over classes.

Based on the object system, a microshell for rapid prototyping of expert systems was built. The efficiency of the compiler integrated approach was proved in an expert system project (Ruckert and Voges 1989).

The Prolog kernel of PROLOG-XT is based on the compiler system SEPIA V2.0 (Meier 1988) of ECRC (European Computer Industry Research Centre). The extensions concerning finite algebras, graphic interface, and definite-clause grammars were designed and implemented at Siemens; the extensions concerning object-oriented programming are based on an implementation of ECRC. It was extended at Siemens with a browser and an external representation formalism. PROLOG-XT runs on the Siemens workstation WS 30-4xx (Apollo DN 4000) under Unix System V. Figure 1, from the domain of circuit design, shows how a full adder is specified and verified in PROLOG-XT.

A circuit verification environment (CVE), based mainly on the extended

(continued on page 26)

. . . artificial neural networks . . . were introduced as highly connected networks of elementary processing elements.

unification and the graphic package of PROLOG-XT, was built. With this tool, real circuits are symbolically verified on gate and switch level. For example, a 16-bit full adder (900 switches) was verified within 12 seconds.

For the first time ever (as far as we are aware), a complete 64-bit ALU (11000 transitors) was symbolically verified. A translator was developed as part of CVE. It maps the internal circuit descriptions of different CAD systems into PROLOG-XT programs, thus providing an interface to conventional design systems.

Two other applications in the domain of digital circuit design are handled by the extended unification: the generation of test patterns and synthesis problems (Tiden 1988). CVE is in use in a design center within Siemens for application-specific integrated circuits (ASICs). Substituting formal verification for lengthy testing and simulation reduces development time and improves reliability. This substitution is of high importance for cost-effective circuit development. The results obtained to date with CVE are promising (Tiden 1989).

We have started using PROLOG-XT to tackle tasks that occur in the design of communication systems. As a first project, a specification and verification environment for communication protocols was developed.

Participants: W. Buettner, K. Estenfeld, R. Enders, H. Leiss, M. Ruckert (until June 1989), R. Schmid, H.-A. Schneider, D. Taubner, and E. Tiden.

Design and Analysis of Neural Networks

Inspired by the functioning of human (or biological) nervous system perception, artificial neural networks (ANNs) were introduced as highly connected networks of elementary processing elements. They exhibit some of the basic characteristics of human perception, such as massive parallelism, adaptivity, and robustness. These properties happen to correspond to principal weaknesses in conventional information processing. Therefore, the recently reactivated field of neurocomputing has drawn enormous scientific and commercial attention to its activities. Neurocomputing is expected to augment conventional computing with an important new paradigm.

To obtain reliable statements about the medium-term potential of ANNs for information technology, Siemens decided to launch a three-year Neurodemonstrator Project (Buettner et al. 1989) to integrate and extend previous activities in neural nets. The project started in October 1988; the total effort involved will be approximately 60 person-years.

The project goal is a system that combines conventional and neural approaches to industrial scene analysis. Methods, principles, and procedures for the design and analysis of ANNs are to be studied and developed under the real-world constraints of the application. Application-specific requirements must be translated into network characteristics. On the software side, a software development environment comprising a compiler for a programming language for ANNs and a comfortable user interface will be provided. For fast emulation of neural nets, a coprocessor board will be developed. The Neurodemonstrator Project is accompanied by research on the technological aspects of ANNs. The Knowledge Processing Laboratory is an important part of this project. The lab's primary effort is in the area of design and analysis of neural networks.

The design of a neural net requires transforming the structure of a problem into the topological-functional structure of a net, which solves the problem in a highly parallel fashion. This mapping is achieved by fixing a connectivity structure; defining input-output nodes; and choosing net parameters such as weights, thresholds, activation functions, and time scales.

In the simplest case, you explicitly extract the topology and parameters for a suitable net from the problem. Various optimization problems, such as the traveling salesman or associative access, can be coded into quadratic (or higher-order) forms, defining a Hopfield net or a Boltzmann machine, that solve the task. It is the massive parallelism that neural nets focus on such problems that appears promising and deserves attention within the project.

In the most difficult case, the structure of a problem is entirely hidden in the data, and the learning must reveal this structure and synthesize a net. As an example of this type of problem, we are currently experimenting with time series derived from biosignals and economic data. Under certain application-specific hypotheses, the results obtained to date are encouraging.

However, both theory and practice confirm that unrestricted learning is costly. To remove part of the burden from the learning routine, explicitly known information about the structure of the problem should be used to construct the coarse structure of the network. One problem of this type is visual pattern recognition subject to certain invariance properties such as rotation. Such invariances induce relations between net parameters, thereby reducing the number of free parameters.

We are currently incorporating such knowledge into the learning routines of various models. It is in this spirit that we expect to perhaps bridge the gap between the symbolic and the connectionist approaches to AI.

We are also exploring another approach to solving invariant pattern-recognition problems in which a vector of geometric features is extracted from a pattern by a preprocessor and then fed to a neural net for recognition.

The prominent feature of the Hopfield net is the existence of a cost function (determined by weights and thresholds) on the state space of the net. Alterations (such as weight changes) in the state of the net should only come about in a cost-reducing manner. Stability, in particular, is ensured this way.

A basic requirement in the Hopfield net, as well as in the other dynamically stable networks reported to date in the literature, is that the connection matrix be symmetric. We have been able to show that stability is preserved under general conditions. Initial results indicate that our generalization is more efficient for pattern association than continuous Hopfield, Cohen-Grossberg, and Kosko models (Schuermann 1989).

An important goal of network design is the development of a methodology that allows nets to be hierarchically specified and composed from subsets, as in circuit design. For simple binary nets, we experiment with PROLOG-XT as an executable net specification language.

We are also developing a network simulator in the form of a C library containing a variety of neural network algorithms and models; a pilot version of the simulator is available. Further work will go into the development of a C-based programming language for neural nets. A compiler will be available in September 1990.

In a later stage of the project, we will need techniques for observing network dynamics, which go beyond panning and zooming. It might be necessary to abandon the microscopic picture of individual nodes and weights and switch to macroscopic descriptions of the network's behavior, for example, by calculating moments of distribution functions for the nodes and weights.

Participants: W. Buettner, F. Hergert, M. Hoehfeld, I. Leuthaeusser, B. Schuermann, M. Weick, P. Witschel, and H. G. Zimmermann.

Bibliography

Block, H. U., and Haugeneder, H. 1988. An Efficiency-Oriented LFG-Parser. In *Natural Language Parsing and Linguistic Theory,* eds. U. Reyle and C. Rohrer. Dordrecht.

Block, H. U., and Haugeneder, H. 1986. The Treatment of Movement-Rules in a LFG-Parser. In Proceedings of the Eleventh International Conference of Computational Linguistics (COLING-86). Bonn.

Block, H. U., and Hunze, R. 1986. Incremental Construction of C- and F-Structure in a LFG-Parser. In Proceedings of the Eleventh International Conference of Computational Linguistics (COLING-86). Bonn.

Block, H. U., and Schlereth, D. 1987. Natural Language Access to SESAM. In Proceedings of the Siemens User's Association. Heidelberg.

Buettner, W. 1989. Implementing Complex Domains of Application in an Extended Prolog System. In *International Journal of General Systems* 15:129–139.

Buettner, W., et. al. 1989. Artificial Neural Networks in Microelectronics. In *Microelectronics for Artificial Neural Nets,* eds. H. Klar and U. Ramacher. Duesseldorf, Germany: VDI.

Buettner, W. 1988. Unification in Finite Algebras Is Unitary(?). In Proceedings of the Ninth International Conference on Automated Deduction. Argonne.

Buettner, W.; Estenfeld, K.; Schmid, R.; Schneider, H.-A.; and Tiden, E. 1988. A General Framework for Symbolic Constraint Handling in Prolog, Technical Report, INF2-ASE-1-88, Siemens AG.

Buettner, W., and Simonis, H. 1987. Embedding Boolean Expressions into Logic Programming. In *Journal of Symbolic Computation* 4:191–205.

Decker, R. 1988. Modeling the Temporal Behavior of Technical Systems. In Proceedings of the Twelfth German Workshop on Artificial Intelligence, Informatik-Fachberichte. Berlin: Springer.

Dressler, O. 1988a. Extending the Basic ATMS. In Proceedings of the Eighth European Conference on Artificial Intelligence. London: Pitman.

Dressler, O. 1988b. An Extended Basic ATMS. In Proceedings of the Second Internal Workshop on Nonmonotonic Reasoning. Berlin: Springer.

Dressler, O., and Farquhar, A. 1989. Problem Solver Control over the ATMS. In Proceedings of the Thirteenth German Workshop on Artificial Intelligence, Informatik-Fachberichte. Berlin: Springer. Also 1989. Technical Report, INF2-ARM-13-89, Siemens AG.

Dressler, O., and Freitag, H. 1989. Propagation of Temporally Indexed Values in Multiple Contexts. In Proceedings of the Thirteenth German Workshop on Artificial Intelligence, Informatik-Fachberichte. Berlin: Springer.

Enders, R. 1989. The Object-Oriented Extension of PROLOG-XT, Technical Report, INF2-ASE-7-89, Siemens AG.

Estenfeld, K., and Ruckert, M. 1988. Knowledge Processing with PROLOG-XT. In Proceedings of Artificial Intelligence in Practice. Munich: Siemens AG.

Farquhar, A. 1989. Modifying the Model Sets during Diagnosis. In Proceedings of the Thirteenth German Workshop on Artificial Intelligence, Informatik-Fachberichte. Berlin: Springer.

Farquhar, A. 1988. A Qualitative Reasoning Approach to Fault Avoidance. In Proceedings of the Eighth European Conference on Artificial Intelligence. London: Pitman.

Frederking, R. E., and Gehrke, M. 1988. Resolving Anaphoric References in a DRT-Based Dialogue System. In Proceedings of the Fourth Austrian Conference on Artificial Intelligence, Vienna Workshop on Knowledge-Based Natural Language Processing, Informatik-Fachberichte. Berlin: Springer.

Freitag, H., and Reinfrank, M. 1988. A Non-Monotonic Deduction System Based on (A)TMS. In Proceedings of the Eighth European Conference on Artificial Intelligence. London: Pitman.

Gehrke, M., and Block, H. U. 1986. Morpheme-Based Lexical Analysis. In Proceedings of the Second Annual Conference of the University of Waterloo Centre for the New Oxford English Dictionary, Advances in Lexicology. Waterloo.

Haugeneder, H., and Gehrke, M. 1988. Improving Search Strategies: An Experiment in Best-First Parsing. In Proceedings of the Twelfth International Conference of Computational Linguistics (COLING-88). Budapest.

Haugeneder, H., and Gehrke, M. 1987. Modeling Heuristic Parsing Strategies. In Proceedings of the Eleventh German Workshop on Artificial Intelligence, Informatik-Fachberichte. Berlin: Springer.

Haugeneder, H., and Gehrke, M. 1986. A User-Friendly ATN Programming Environment (APE). In Proceedings of the Eleventh International Conference of Computational Linguistics (COLING-86). Bonn.

Huberman, B., and Struss, P. 1989. Chaos, Qualitative Reasoning, and the Predictability Problem. In Proceedings of the 1989 Qualitative Reasoning Workshop. Palo Alto.

Hunze, R. 1988. Resolving Anaphoric References in a DRT-Based Dialogue System: Part 1, The Formal Discourse Model, WISBER Report 24, Saarbruecken Univ.

Meier, M., ed. 1988. SEPIA Version 2.0, User Manual, Technical Report, European Computer Industry Research Centre.

Reinfrank, M. 1989. Logical Foundations of Nonmonotonic Truth Maintenance. In Proceedings of the Fourth Portugese AI Conference, ed. J. Martins. Berlin: Springer.

Reinfrank, M., and Dressler, O. 1989. On the Relation between Truth Maintenance and Non-Monotonic Logics. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Reinfrank, M., and Freitag, H. 1988. Rules and Justifications: A Uniform Approach to Reason Maintenance and Non-Monotonic Inference. In Proceedings of the International Conference on Fifth Generation Computer Systems. Tokyo: Institute for New Generation Computer Technology.

Ruckert, M., and Voges, U. 1989. WIFEX—WInterweizen Fungizid EXperte, Technical Report, INF2-ASE-6-89, Siemens AG.

Schachtl, S. 1988. The Problem of Overgeneration in Parsing Processes and the Aid of Linguistic Generalizations. In Proceedings of the Fourth Austrian Conference on Artificial Intelligence, Vienna Workshop on Knowledge-Based Natural Language Processing, Informatik-Fachberichte. Berlin: Springer.

Schmid, L. 1988. Case-Driven Extension of the Lexicon for Verbs in SEPP, Internal Report, Siemens AG.

Schuermann, B. 1989. Generalized Adaptive Bidirectional Associative Memory. In Proceedings of the International Neural Network Conference. San Diego.

Struss, P. 1989a. Diagnosis as a Process. In Proceedings of the 1989 Workshop on Model-Based Diagnosis. Paris: IBM.

Struss, P. 1989b. Model-Based Reasoning, Progress, and Problems. In Proceedings of the 1989 Congress of the German National Computer Science Association (GI Congress) on Knowledge-Based Systems. Berlin: Springer.

Struss, P. 1988a. A Framework for Model-Based Diagnosis, Technical Report, INF2-ARM-10-88, Siemens AG.

Struss, P. 1988b. Extensions to ATMS-Based Diagnosis. In *Artificial Intelligence in Engineering: Diagnosis and Learning*, ed. J. Gero, 3–28. Southampton, England: Computational Mechanics.

Struss, P. 1988c. Global Filters for Qualitative Behaviors. In Proceedings of the Seventh National Conference on Artificial Intelligence. Menlo Park, Calif.: American Association for Artificial Intelligence.

Struss, P. 1988d. Mathematical Aspects of Qualitative Reasoning. In *International Journal of Artificial Intelligence in Engineering* 3:156–169.

Struss, P. 1988e. Mathematical Aspects of Qualitative Reasoning—Part Two: Differential Equations, Technical Report, INF2-ARM-7-88, Siemens AG.

Struss, P. 1987. Multiple Representation of Structure and Function. In *Expert Systems in Computer-Aided Design*, ed. J. Gero. Amsterdam: North-Holland.

Struss, P., and Dressler, O. 1989. Physical Negation—Integrating Fault Models into the General Diagnostic Engine. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Tiden, E. 1989. Rapid Development of VLSI CAD Tools with PROLOG-XT, Technical Report, INF2-ASE-4-89, Siemens AG.

Tiden, E. 1988. Symbolic Verification of Switch-Level Circuits Using a Prolog Enhanced with Unification in Finite Algebras. In Proceedings IFIP WG 10.2 Conference. Glasgow.

Wolfram Buettner received his diploma in mathematics from the University of Heidelberg in 1975 and a Ph.D. in mathematics from Tulane University in 1978. In 1985, he joined Siemens Corporate Research, where he currently heads the Knowledge Processing Laboratory. His research interests include the design of Prolog extensions for circuit design and algorithmic aspects of neural networks. He is also affiliated with the University of Kaiserlautern, where he is a professor of computer science.

Klaus Estenfeld is leader of the Language Extension Group at Siemens Corporate Research and Development, with special interests in Prolog implementation and the extension of Prolog for special applications (for example, computer-aided design electronics, XPS). He received his diploma in 1976 and his Ph.D. in 1980, both in computer science from the University of Saarbrucken. In 1981, he joined the Siemens Research Labs, working in the field of program construction and verification, with special interests in parsing and unification. From 1984 to 1986, he was project leader in the Logic Programming Group of the European Computer-Industry Research Center (ECRC), where he was responsible for the ECRC Prolog Project.

Hans Haugeneder is co-leader of the Natural Language Technology Group at Siemens Corporate Research and Development. He holds a Master's in linguistics and logics from the University of Muenchen for his work on extensions of Montague semantics. His key interests in natural language understanding focus on parsing models, descriptive formalisms for linguistic knowledge sources, and domain-independent language processing systems. Currently, he is at the German Research Center for AI in Saarbrucken, where he is leading a project on human-computer cooperation and communication in distributed environments.

Peter Struss is leader of the Advanced Reasoning Methods Group at Siemens Corporate Research and Development. He received his Ph.D. in computer science from the University of Kaiserlautern for his research on qualitative reasoning with a focus on mathematical formalization and analysis of qualitative calculi. His key interests are model-based reasoning and the integration of qualitative reasoning and non-monotonic reasoning, including problems such as the relations between qualitative modeling and default reasoning in diagnostic systems, the role of commonsense concepts in engineering problem solving, and the interaction between model-based and experience-based diagnostic reasoning. His current work is aimed at foundations for model-based diagnostic systems that can deal with significant real applications.