Research in the Computer Science Department at Indiana University 1980-81

edited by

Mitchell Wand

Computer Science Department

Indiana University

Bloomington, Indiana 47405

TECHNICAL REPORT NO. 122

RESEARCH IN THE COMPUTER SCIENCE DEPARTMENT

at INDIANA UNIVERSITY

1980-81

Edited by

Mitchell Wand

December, 1981

1. Faculty

John Barnden, Assistant Professor; Ph.D., Oxford. Artificial intelligence, programming languages, formal semantics.

Cynthia Brown, Assistant Professor; Ph.D., University of Michigan. Analysis of algorithms, compilers.

John Buck, Lecturer and Assistant Chairman; B.S., Virginia Tech. Systems analysis, data base and information systems, computer science education.

James Burns, Assistant Professor; Ph.D. Georgia Tech. Theoretical computer science, parallel and distributed systems, distributed data bases.

Will Clinger, Assistant Professor; Ph.D., MIT. Semantics of programming languages, artificial intelligence, nondeterministic concurrent computation, logic.

George Epstein, Professor; Ph.D., University of California, Los Angeles. Systems design, multiple-valued logic, computer science education.

Robert Filman, Assistant Professor; Ph.D., Stanford University. Artificial Intelligence, programming languages, distributed computing, data base systems, logic, and man-machine interaction.

Daniel Friedman, Associate Professor; Ph.D., University of Texas at Austin. Program methodology, formal semantics, artificial intelligence, distributed computing, simulation languages, applicative languages, LISP.

Stanley Hagstrom, Professor, Computer Science and Chemistry; Ph.D., Iowa State University. Computer hardware, laboratory automation, computer networking, operating systems, software engineering, analysis of algorithms in ab initio quantum chemistry.

Smith Higgins, Associate Professor; Ph.D., Notre Dame. Computer science education, number theory, modern algebra.

Douglas Hofstadter, Associate Professor; Ph.D., University of Oregon. Artificial intelligence, perception of patterns and style, self-monitoring and self-perception, philosophy of mind.

Stan Kwasny, Assistant Professor; Ph.D., Ohio State University. Natural language understanding, artificial intelligence, data structures, computational linguistics, data base systems.

John O'Donnell, Assistant Professor; Ph.D., University of Iowa. Computer architecture, operating systems, VLSI design, programming languages.

Franklin Prosser, Professor; Ph.D., Pennsylvania State University. Digital hardware, operating systems, computer science education.

Paul Purdom, Associate Professor and Chairman; Ph.D., California Institute of Technology. Analysis of algorithms, compilers.

Edward L. Robertson, Associate Professor; Ph.D., University of Wisconsin. Theory of computation, computational complexity, hardware and software systems architecture, data bases.

Mitchell Wand, Associate Professor; Ph.D, Massachusetts Institute of Technology. Theory of programming, logic, algebra.

David Wise, Associate Professor; Ph.D. University of Wisconsin. Applicative programming, data structures, multiprocessing architectures and languages.

2. Facilities

During 1980-81, the department acquired over half a million dollars of new equipment. The major acquisitions include two large minicomputers, a Hewlett-Packard 3000 model 44 and a VAX 11/780 from DEC. Additional equipment acquired during 1981-82 include communication gear and almost 50 more terminals (including five color graphic terminals).

The 3000/44 is a gift of Hewlett-Packard to help provide instructional computing, primarily in sophomore and junior-level classes. It is configured with 1M of primary memory and 240M of disk, which should provide ample storage space for students' files. It will have ports for 24 terminals (although initially there may be fewer attached), a printer and a magnetic tape for backup.

The VAX was primarily funded by a grant from the National Science Foundation, with substantial University support, to be used by faculty and graduate students for research (and incidental tasks such as preparing this document). Running the sophisticated UNIX operating system, the VAX is already a popular machine generating many requests for projects on it.

We replaced our aging TI 980's by a laboratory of fifteen 6809 microcomputers. Each system has dual floppy disk units and a CRT, with a simple operating system and debug monitor. This will enable our assembly language classes to do their work in "hands on" mode. The operating systems class will use five LSI-11's given by Digital Equipment Corp.

Additional computing facilities for the Computer Science Department are provided by the Indiana University Computing Network. This network currently consists of a CDC 6600/CYBER 172 and a PRIME 750 in Bloomington and a DEC KL10 and an IBM 4341 in Indianapolis. A VAX 11/780 will be installed in January 1982. There is limited network access to an IBM 370/158 in Bloomington, used primarily for administrative computing. Student access to University facilities has improved with a departmental terminal cluster, which shares a Lindley Hall room with a cluster of terminals hard-wired to a University Prime 750 and will further improve thanks to $6.5 million provided by the state.

The department maintains a well-equipped hardware development laboratory, with an assortment of microprocessors and supporting equipment. A feature of this laboratory is the Logic Engine, a general-purpose controller for horizontally microprogrammed systems. This device has been used to build disk controllers and small computers; it is currently being used in the development of a FORTH machine. A second feature is a facility for the production of two-sided plated-through printed-circuit boards, such as the one used for the Logic Engine.

Plans for 1982 include making the VAX a telephone host on CSNet, and interconnecting the computing resources in Lindley Hall with an Ethernet. An Imagen laser printer is also on order.

3. Summaries of Research Projects

John A. Barnden

An Imagery Model of Symbolic Cognition

An unconscious-imagery model of symbolic cognition is under investigation in an artificial intelligence project. In the model, temporary symbolic structures appear as imaginal diagrams analogous to real drawings of the structures. The diagrams are states of abstract array-like data structures which are also used to hold unconscious enhanced images resulting from front-end perceptual processing. The present study is attempting to elucidate the expressive and problem-solving convenience of the unusual (e.g. hybrid pictorial/abstract) symbolisms natural to the model. The project also involves the computer simulation of the necessary diagram-manipulation processes.

Concurrent Programming Using Phases

A concurrent programming language, PhasL, is being developed in which the notions of condition and process are united in a single 'phase' concept, and in which there are no sequential control constructs. All actions are produced by guarded commands, where a guard detects the presence and absence of phases and where a satisfied guard triggers an unstructured set of hase-manipulation operations. PhasL allows elegant solution of complex concurrent-programming problems. There are some interesting implementational and formal-semantical problems for which solutions are being devised. Automatically checkable restrictions of the language to allow distributed implementation are being studied. PhasL is a subset of a combined continuous-discrete simulation language, PhasL/SIM, which was developed earlier. The possibility of extending PhasL/SIM to a modular-production-system language for artificial-intelligence applications is receiving attention.

James E. Burns
John O'Donnell
Franklin Prosser
Edward L. Robertson

<u>Simulation Studies of Distributed Systems</u>

Distributed systems are becoming ever more significant with advancing technology, but analysis is made impossible by the complex interactions that occur. The proposed research would develop simulation tools for distributed systems and apply these tools to the study of particular aspects of these systems. The tools would allow specifying systems in detail or by stochastic parameters - for example, specifying actual message content versus average message traffic - and would facilitate clustering the behavior of subsystems as units in a larger system. Particular topics to be considered included distributed data bases, load sharing systems, and algorithms providing synchronization and security.

See also: Edward L. Robertson

Robert E. Filman

<u>Meta-Reasoning in Knowledge Representation</u>

A central issue in the creation of an intelligent computer system over a domain is the representation of the knowledge embedded in that domain in machine-usable forms. This research suggests that it is possible to construct intelligent systems where the underlying facts, the legal conclusions derivable from those facts, and the appropriate control strategy to derive those conclusions can all be expressed in machine-usable form as instances of meta-theoretic reasoning. We are exploring the expression and resolution of knowledge representation problems by a combination of procedural attachment, rewriting systems, and meta-theoretic representations, in the context of a first order logic system. Issues to be addressed include the representation of heuristics and control, hypothetical reasoning, non-monotonic systems, knowledge and belief, time, strategies, learning and game theory.

Robert E. Filman and Daniel P. Friedman

<u>Models and Techniques for Distributed Systems</u>

Currently we are writing a book based on the development of tools and techniques for distributed software. In the text we describe and discuss some 15 languages and models for interprocess communication. In addition, we discuss heuristics for problem solving in the distributed processing environment.

See also: Friedman and Wise
Wand and Friedman
Wise and Friedman

Daniel P. Friedman and David S. Wise

Applicative Programming for Systems

This project extends an applicative approach to programming parallel computers for system-level problems which require synchronization. This applicative style requires that all programs be stated as expressions in a calculus, where arguments are input and values are output; sequence of execution or evaluation is irrelevant. Synchronization of a process with an external event--perhaps the output of another processor--is accomplished through the multiset data structure, which encapsulates all the interprocessor contention within a shared data object. We have developed the semantics of multisets and lists at the formal level, but several issues stand between this approach to difficult system-level problems (e.g. airline reservation system) and its direct implementation on arbitrary numbers of communicating processors.

The most important one is fairness. An originally unordered multiset takes on an order as it is probed, but it may also be augmented; fairness requires that convergent values, whether original or augmenting, may not be deferred arbitrarily in that order. Another problem is controlling the propagation of the strict functions which we have introduced in order to restrict convergence of structured values within a multiset. The selection of a storage-management scheme for a heap shared by many processors remains open; generalizations to reference counting or restrictions to garbage collection are needed to constrain memory conflicts. Several other problems, including the role of functions as values, and the strategies for firmware implementation of multisets, will be considered.

See also: Friedman and Filman
Wand and Friedman
Wise and Friedman

Douglas R. Hofstadter

<u>SW</u>: <u>A Computer Model of Perception</u>, <u>Abstraction</u> & <u>Induction</u>

"SW" is the name for the common structure underlying two artificial intelligence research projects: Seek-Whence and See-Write. Our goal is to make a program that is capable of perception of patterns composed of integers. The patterns involve complex structures rather than mathematical properties. Seek-Whence, by looking at initial segments of a sequence, tries to find rules for it, using ideas of evidence, plausibility, and simplicity. For See-Write, the goal is to infer, from a few alphabetic characters drawn for it on a terminal's screen, a stylistic quality, and then to extend that quality, by an analogical transfer, to other letters, thus coming up with a complete typeface in a style that has been partially specified through examples. Our overall aim in SW is to implement a single, unified program that can "maneuver" descriptions of objects into abstract and functionally defined categories (i.e.,recognize objects), perceive HOW those objects fit into their categories, (i.e., perceive their style), and create new, fluid categories out of old ones (i.e., learn new concepts).

Stan C. Kwasny

__Interpreting Errorful Inputs in Natural Language Understanding Systems__

We are investigating how techniques designed for interpreting syntactically ill-formed inputs in a Natural Language Understanding System can be applied to other stages of language. Specifically addressed will be the question of whether these or other techniques can operate at the semantic and pragmatic levels to achieve a more robust understanding at those levels. When this is achieved, the overall robustness of the language processing component of such a system will be enhanced.

In this study, a Cascaded Augmented Transition Network (CATN) model of grammar is utilized. This permits interaction between several Augmented Transition Network (ATN) grammars written for various stages of language processing. It is our theory that malformations realized at the surface or syntactic level may also occur at other levels as well. Furthermore, the nature of these deeper malformations is identical in many ways to the nature of malformations that have been successfully treated at the syntactic level. This theory will be tested during the course of this work.

Franklin Prosser

Structured Hardware Design

The theme of my major research interests is to make hardware design processes more systematic, structured, and orderly. This theme has led me down two major avenues: teaching of structured digital hardware design, and the development of design aids to support structured design. In both of these areas I have worked with Professor David Winkel of Univ. of Wyoming. Our interest in teaching digital design led us to develop hardware instructional laboratories at Indiana University and at the University of Wyoming, and we wrote a current textbook (Winkel and Prosser, The Art of Digital Design, Prentice-Hall, Inc.). The labs and the book contribute to my goal of teaching students to deal with complexity in orderly and systematic ways; in the lab each student builds a fully functional minicomputer from MSI-level components, using structured design and debugging techniques.

In our research in structured design, we are developing a Logic Engine -- a system for microprogrammed control of hardware architectures. The system has (a) a powerful base unit for clocks, display, and power, (b) a large printed circuit board containing the Logic Engine controller, a microcomputer system for the user interface, and copious space for the user's hardware design, and (c) a software support system to assist the user in developing and testing microcode and debugging the overall design.

Using the Logic Engine as the control element, we are developing a FORTH machine for high-speed execution of programs written in the FORTH language.

Also, using a simplified version of the Logic Engine, I am revising my hardware instructional laboratory, so that students may produce their minicomputer designs using either hardwired or microprogrammed control elements. I hope to augment the present laboratory with this new teaching capability in Spring 1983.

Soon I hope to develop a VLSI version of the Logic Engine control circuits, since the main portion of the Engine should fit nicely into a VLSI design.

On a more abstract level, I have been working to develop structured design methods and incorporate them into routine hardware design. Among the techniques that we have found most useful are mixed logic for circuit descriptions and the ASM description (after Osborne and Clare) of control algorithms. These techniques form the backbone of our textbook. Recently, I have made some progress in incorporating these and similar techniques into VLSI design.

Paul W. Purdom, Jr. and Cynthia A. Brown

Average Time Complexity of NP-Complete Problems

The worst case complexity of the best known algorithms for solving NP-complete problems is exponential. We are studying the average time complexity of satisfiability using various search algorithms on models of random conjunctive normal form predicates. A class of predicates over v variables is characterized by p(v) , the probability a given literal is in a random clause, and t(v) , the number of random independently selected clauses in a predicate. We are characterizing the functions p(v) and t(v) for which some algorithm takes polynomial average time in the limit as v becomes large. We have found algorithms that take polynomial average time when p(v) or t(v) grow extremely slowly and when t(v) grows quickly compared to p(v) .

Efficient Global Error Recovery

We are studying methods for improving the efficiency of global error recovery in compilers. Good error recovery is fundamental to an effective compiler. Some errors cannot be corrected in a reasonable way without global recovery, but current compilers use local methods because global techniques are too costly. Our approach to improving the efficiency of global methods involves (1) speeding up Earley's algorithm to make it competitive with LR(k) parsing on LR(k) grammars; (2) improving the global error recovery algorithm of Aho and Peterson, which is based on Earley's algorithm, by having it pursue only the cheapest parse(s); (3) exploring ways to limit the number of alternative parses considered by Aho and Peterson's algorithm by using information already present on the parsing stack. We will investigate the application of our results to syntax chart grammars and to natural language processing.

Edward L. Robertson

<u>Studies Related to NP-Complete Problems: Structure Approximation and Backtracking</u>

Important practical problems in the class of "NP-complete problems", from such diverse areas as industrial management and computer network reliability, are all difficult to solve, in the sense that all known general methods for solution are not much better than trying all possible cases. Moreover, these problems are all related, so that an efficient solution method for any one problem would provide efficient solutions for all of them. This research intends to 1) discover properties characterizing NP-complete problems, 2) study when it is possible to find approximate solutions to NP-complete problems given that finding exact solutions is too costly, and 3) develop methods to ignore hopeless cases if a case-by-case search for solutions is indeed necessary.

See also: James E. Burns et.al.

Mitchell Wand and Daniel P. Friedman

Algebraic and Logical Semantics of Computation

Our research in the semantics of computation has focused on the use of continuations and their representations in a variety of contexts. We are extending this investigation to study implementations and representations of denotational semantics. In particular, we are studying compiler correctness with respect to Scott-Strachey semantics. We hope to provide alternatives to the use of so-called congruence relations for this task. We have developed techniques for deriving target code as a representation of continuation semantics for a sizable class of languages. These techniques have implications for the design of language-oriented machines in microcode or VLSI; we hope to explore these implications as well.

See also: Filman and Friedman
Friedman and Wise
Wise and Friedman

David S. Wise and Daniel P. Friedman

Applicative Programming for Indeterminate Systems

Indeterminate systems are operating systems, data base systems, distributed systems, and multiprocesser systems that require real-time response to many independent conditions that occur with a relative synchronization not known at the time the system is built. Applicative or functional programming is a style of expressing computer algorithms as mathematical function definitions, which are specifically devoid of time- and side-effects.

Extending applicative programming to deal with the essential properties of timing in systems is important because it is already so promising for efficient use of these same systems for time-independent tasks. Traditional programming styles do not make good use of parallelism available in new architectures because they have been modelled after single processor computers. They have, however, been extended to deal with synchronization issues. Applicative languages, which can well use parallelism unknown to the programmer, have not been extended to deal with all of the problems in working with such systems.

We propose to develop the DAISY programming language and DSI system to a production environment that can cope with issues of indeterminism among input conditions, breadth-first evaluation of condition trees, failures of output devices, and embedded systems, and to refine system performance with respect to program maintenance and automatic introduction of sequentality to reduce time and space needs.

See also: Filman and Friedman
Friedman and Wise
Wand and Friedman

Table of Contents

4. Publications

a. Publications appearing in 1980-81

Barnden, J. "Nonsequentiality and Concrete Activity Phases in Discrete-Event Simulation Languages," ACM Trans. Prog. Langs. and Systems, 3(1981) pp. 293-317.

Becker, L. "PHONY: A Heuristic Phonological Analyzer," Proc. 19th Ann. Meeting of Assoc. for Computational Linguistics (1981).

Brown, C.A. & Purdom, P.W. "An Average Time Analysis of Back-tracking," SIAM J. Comput. 10 (1981), 583-593.

Burns, J.E. "Mutual Exclusion Using Indivisible Reads and Writes," 18th Allerton Conf. (with N. Lynch). pp. 833-842.

Epstein, G. "Selecting Don't-Care Sets for Symmetric Functions: A Pictorial Approach Using Matrices," Proc. of 10th Int'l Symp. on Multiple-valued Logic (1980) pp. 219-225 (with D. M. Miller and J. C. Muzio).

Epstein, G. "On Rine's View of Boolean Algebras," Proc. of 11th Int'l Symp. on Multiple-valued Logic (1981) pp. 256-258.

Filman, R. E., "Computers and Chess," Mathematical Intelligencer, Spring 1981.

Filman, R.E., and Friedman, D.P., "Inspiring Distribution in Distributed Computation." presented at SIGOPS/SIGPLAN Workshop on Fundamental Issues in Distributed Computing (December, 1980).

Friedman, D.P. "CONCUR: A Language for Continuous, Concurrent Processes," Computer Languages 5 (1980) pp. 163-189 (with R. M. Salter and T. J. Brennan).

Hofstadter, D. "On Self-Referential Sentences," Scientific American (Jan. 1981).

Hofstadter, D. "Rubik's Magic Cube," Scientific American (Mar. 1981).

Hofstadter, D. "The Turing Test: A Coffeehouse Conversation," Scientific American (May, 1981).

Hofstadter, D. "Heisenberg's Uncertainty Principle and the Many-Worlds Interpretation of Quantum Mechanics," Scientific American (July, 1981).

Kwasny, S. "Treatment of Ungrammatical and Extra-Grammatical Phenomena in Natural Language Understanding Systems," Indiana University Linguistics Club, November, 1980.

Kwasny, S. "Relaxation Techniques for Parsing Grammatically Ill-Formed Input in Natural Language Understanding Systems," _American Journal of Computational Linguistics_, Vol. 7, No. 2, 99-108, April-June, 1981 (with N.K. Sondheimer).

Purdom, P.W., Brown, C.A. & Robertson, E. "Backtracking with Multi-Level Search Rearrangement," _Acta Informatica 15_ (1981) pp. 99-113.

Purdom, P.W. & Brown, C.A. "Parsing Extended LR(k) Grammars," _Acta Informatica 15_ (1981) pp. 115-127.

Purdom, P.W. & Brown, C.A. "Semantic Routines and LR(k) Parsers," _Acta Informatica 14_ (1980) pp. 299-315.

Purdom, P.W. & Brown, C.A. "Exact Formulas for the Buddy System," _Information Sciences 22_ (1980) pp. 1-18.

Robertson, E. L. "Continual Pattern Replication" _Information and Control 48_ (1981), pp. 211-220 (with J. I. Munro).

Wand, M. "Continuation-Based Multiprocessing," _Proc. 1980 LISP Conference_, pp. 19-28.

Wand, M. "First-Order Identities as a Defining Language," _Acta Informatica 14_ (1980) pp. 337-357.

Wise, D.S. "Compact Layout of Banyan/FFT networks." In H. Kung, B. Sproull, and G. Steele (eds.), _VLSI Systems and Computations_, Rockville, Md. Computer Science Press (1981).

b. _Articles to appear_

Burns, J.E. "Data Requirements for the Implementation of N-Process Mutual Exclusion Using a Single Shared Variable," _JACM_ (with Lynch, Fischer, Jackson & Peterson).

Burns, J.E. "Symmetry in Systems of Asynchronous Processes," _22nd Conf. on Found. of Comp. Sci._ (1981).

Epstein, G. "Positive Multiple-valued Switching Functions--An Extension of Dedekind's Problem," _Proc. of 12th Int'l Symp. on Multiple-valued Logic_ (with Y. W. Liu).

Epstein, G. "The Underlying Ground for Hypothetical Propositions," _Scientia_.

Epstein, G. "A Summary of 'Core Points in Double Heyting Algebras and Dissectable Lattices," _Polish Acad. of Sci., Bull. of Section of Logic_ (with A. Horn).

Filman, R.E., and Friedman, D.P. "Models, Languages, and Heuristics for Distributed Computing." National Computer Conference 1982. AFIPS Press.

Filman, R.E. & Friedman, D.P. Coordinated Computing: Tools and Techniques for Distributed Software, McGraw-Hill, New York.

Robertson, E.L. "On the Density and Structure of NP-Complete and NP sets," Theor. Comp. Sci., (with L. Landweber and R. Lipton).

Wand, M. "Specifications, Models, and Implementations of Data Abstractions," Theoretical Computer Science.

Wand, M. "Target Code as a Representation of Continuation Semantics," ACM Trans. on Prog. Lang. and Systems.

Wand, M. "Semantics-Directed Machine Architecture", Conf. Rec. ACM Symp. on Principles of Programming Languages (1982).

Wise, D. S. "Interpreters for functional programming." In J. Darlington, P. Henderson, and D. Turner (eds.), Functional Programming and its Applications, Cambridge University Press (1982).

Wise, D. S. "Functional Programming." In A. Ralston (ed.), Encyclopedia of Computer Science, New York, Van Nostrand (1982).

c. Books in Print

Epstein, G. Modern Uses of Multiple-valued Logic, Reidel, 1977 (Co-editor with J. M. Dunn).

Friedman, D.P. The Little LISPer, Science Research Associates, Palo Alto, 1974.

Hofstadter, D. R. Godel, Escher, Bach: an Eternal Golden Braid, Basic Books, New York, 1979.

Hofstadter, D.R. and Dennett, D. (eds.) The Mind's I, Basic Books, New York, 1981.

Kreitzberg, C.B. & Shneiderman, B. Fortran Programming: A Spiral Approach, Harcourt Brace Jovanovich, New York, 1975.

Shapiro, S.C. Techniques of Artificial Intelligence, Van Nostrand, New York, 1979.

Wand, M. Induction, Recursion, and Programming, Elsevier North Holland, New York, 1980.

Winkel, D. & Prosser, F. The Art of Digital Design: An Introduction to Top-Down Design, Prentice-Hall, Englewood Cliffs, N. J., 1980.

d. Other Technical Reports

Wolynes, G., Ginder, J., Roitblat, B. and Roland, V. "An Experiment in SCHEME-Based Distributed Processing," Technical Report No. 97 (August, 1980).

Wand, M. (ed.), "Research in the Computer Science Department at Indiana University," Technical Report No. 98 (August, 1980).

Filman, R. and Friedman, D.P. "Inspiring Distribution in Distributed Computation," Technical Report No. 99 (December, 1980).

Brown, C.A. and Purdom, P.W. Jr. "An Empirical Comparison of Backtracking Algorithms," Technical Report No. 100 (December, 1980).

Purdom, P.W. Jr. and Brown, C.A. "The Goldberg Putnam-Davis Procedure Requires Exponential Average Time," Technical Report No. 101 (January, 1981).

Epstein, G. "Context-free Grammars for the Balanced, Overbalanced and Unbalanced Binary Trees". Technical Report No. 103 (March. 1981)

Brown, C.A. and Purdom, P.W. Jr. "How to Search Efficiently," Technical Report No. 105 (March, 1981).

Dwyer, R.A. and Dybvig, R.K. "A SCHEME for Distributed Processes," Technical Report No. 107 (April, 1981).

Grismer, T.M. "Solving Common Programming Problems with an Applicative Programming Language," Technical Report No. 109 (June, 1981).

Clossman, G. "The SKI Manual," Technical Report No. 110 (July, 1981).

Brown, C.A. and Purdom, P.W. Jr. "Specifying One-Pass Bottom-Up Compilers," Technical Report No. 113 (July, 1981).

5. Colloquium Series

Luigia Aiello
Istituto di' Elaborazione delle Informazione
Using meta-theoretic reasoning to do elementary algebra in FOL
September 2, 1980

Paul W. Purdom, Jr.
Indiana University
Parsing Extended LR(k) Grammars
September 9, 1980

Steven D. Johnson
Indiana University
An Intuitive look at FRONS
September 16, 1980

John A. Barnden
Indiana University
A Style of Language which Structure Systems as Activity Phase Nests
October 7, 1980

Nachum Dershowitz
University of Illinois
Proving Termination of Term-rewriting Systems
October 14, 1980

Peter Hibbard
Carnegie-Mellon University
Spice - An Experiment in Personal Scientific Computing
October 24, 1980

Laurent Kott
University of Paris VII
Program Transformations
November 6, 1980

Sam Kamin
University of Illinois
The Abstract Specification of Abstract Data Types
November 11, 1980

Walter L. Ruzzo
University of Washington
A Computational Complexity Theory for Highly Parallel Computers?
November 13, 1980

Keith Clark
Imperial College London
IC-Prolog
November 18, 1980

Mitchell Wand
Indiana University
Target Code as Syntactic Vinegar
December 2, 1980

Edward Robertson
Indiana University
The VAX Computer Facility
February 3, 1981

Tim Teitelbaum
Cornell University
The Cornell Program Synthesizer: A Syntax-Directed Programming Environment
February 19, 1981

Robert Lewis (Scot) Drysdale, Ill
Dartmouth College
Generalized Voronoi Diagrams
February 27, 1981

Alan Selman
Iowa State
Reducibilities, Relativization, and Relationships between Complexity Classes
March 2, 1981

Raymond Smullyan
I.U. Philosophy & C.C.N.Y.
Fixed Points & Self Reference
March 5, 1981

Lee Becker
Indiana University
PHONY: A Heuristic Phonological Analyzer
March 12, 1981

William Clinger
MIT
Foundations of Actor Semantics
March 13, 1981

Carl H. Smith
Purdue University
A General Theory of Automatic Program Synthesis
March 18, 1981

Michael Gemignani
Indiana-Purdue at Indianapolis
Thoughts on Computer Related Law
March 31, 1981

Fanya S. Montalvo
Hewlett Packard Laboratories
Human Vision, Computer Graphics and Other Illusions
April 2, 1981

Gyorgy E. Revesz
University of Kentucky
Lambda-Calculus and the Semantics of Programming Languages
April 7, 1981

Marilyn Mantei
University of Michigan
Disorientation Behavior in Person-Computer Interaction
April 10, 1981

Vincent A. Mabert
Indiana University
Static vs. Dynamic Priority Rules for Check Processing in
  Multiple Branch Banking
April 14, 1981

Stuart C. Shapiro
SUNY at Buffalo
Semantic Network and Intensional Concepts
April 16, 1981

Raymond Smullyan
I.U. Philosophy & C.C.N.Y.
This Talk has no Title
April 23, 1981

Donna Brown
University of Illinois
A General Class of Resource Tradeoffs
April 28, 1981

Irene Guessarian
Washington State University
Algebraic Semantics of Program Schemes
May 19, 1981