Artificial Intelligence Project
at
The University of Texas at Austin

Proposal to Army Research Office
AI-TR-84-1

22 November 1983

Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712

Principal Investigators:
Gordon S. Novak Jr.
Department of Computer Sciences
(512) 471-4353

Robert L. Causey, Chairman
Department of Philosophy
(512) 471-4857
This report contains the technical part of a proposal that was submitted to the U.S. Army Research Office (Electronics Division, Dr. Jimmie R. Suttle, Director) in late 1983 in response to their call for proposals on Artificial Intelligence Research and Education. The University of Texas at Austin (along with the University of Pennsylvania) was selected for substantial funding over a five-year period out of a total of thirty-five proposals submitted. This report also contains a subsequent proposal by Dr. Bruce Porter.

The individual proposals in this document illustrate the breadth of research in Artificial Intelligence at the University of Texas, though not all of the proposed projects were selected for funding. The research projects that are currently supported by the Army grant are those of Novak, Simmons, Kumar, and Porter.
# Table of Contents

1. Introduction  
   1.1 The University of Texas at Austin  
   1.2 Artificial Intelligence at U.T. Austin  
   1.3 Areas of Strength in Artificial Intelligence  
      1.3.1 Problem Solving  
      1.3.1.1 İSAAC Program  
      1.3.1.2 Problem Solving Search Algorithms  
      1.3.1.3 Parallel Algorithms  
      1.3.1.4 Architectures for Problem Solving Search  
      1.3.1.5 Problem Solving under Uncertainty  
   1.3.2 Software Tools for A.I.  
      1.3.2.1 GLISP  
      1.3.2.2 HCPRVR  
      1.3.2.3 Theorem Provers  
   1.3.3 Mechanical Theorem Proving  
      1.3.3.1 References  
   1.3.4 Natural Language Processing  
   1.3.5 Cognitive Science  
   1.3.6 Automated Translation  
   1.3.7 Intelligent Interfaces  
   1.3.8 Laboratory for Image and Signal Analysis  
   1.3.9 Robotics  
1.4 Research Computer Facilities  
   1.4.1 Computer Sciences Facilities  
   1.4.2 Flexible Automation/Robotics Laboratory  
   1.4.3 University Facilities  

2. Research  
   2.1 Introduction  
   2.2 Research Structure  
   2.3 Extensions to GLISP  
      2.3.1 Introduction  
      2.3.2 Data Description Language  
      2.3.3 Abstract Algorithms  
      2.3.4 References  
   2.4 Expert Problem Solving in Physics  
      2.4.1 Introduction  
      2.4.2 Man-Machine Communication  
      2.4.3 Representation of Knowledge  
      2.4.4 Control of the Problem-Solving Process  
      2.4.5 Applications  
      2.4.6 Research Plan  
      2.4.7 References
2.5 Device Descriptions in the Text Knowledge System 22
  2.5.1 References 23

2.6 Intelligent Interfaces 24
  2.6.1 The Scribe Tutor 24
  2.6.2 The Programming Tutor 25
  2.6.3 The Intelligent Operating System Interface 26
  2.6.4 References 26

2.7 Problem Solving under Uncertainty 28
  2.7.1 References 28

2.8 Research and Development in Automated Theorem Proving 29
  2.8.1 Introduction 29
  2.8.2 Current Research 29
  2.8.3 References 31

2.9 The Mechanical Derivation of Concurrency in Programs 34

2.10 Integrating Knowledge into Problem Solving Search Procedures 37
  2.10.1 Introduction 37
  2.10.2 Previous Work 38
  2.10.3 Proposed Research 38
  2.10.4 References 40

2.11 Parallel Algorithms and Architectures for Problem Solving Search 42
  2.11.1 Review of Previous Work 42
  2.11.2 Further Work on Design of Parallel Algorithms 42
  2.11.3 Interprocess Communication vs. Speedup 43
  2.11.4 Special-purpose Architectures for Problem Solving 43
  2.11.5 References 44

2.12 Discourse Representation and Processing 47
  2.12.1 Semantics 48
  2.12.2 Syntax 49
  2.12.3 References 50

2.13 Artificial Intelligence and Robotic Manipulators 52
  2.13.1 Introduction 52
  2.13.2 System Design and Components 55
  2.13.3 References 56
  2.13.4 Equipment Needs 57

2.14 AI in Target Recognition and Tracking 60
  2.14.1 References 61

2.15 Automated Translation and Natural Language Processing 62
  2.15.1 Introduction 62
  2.15.2 Past Linguistic Research Methodology 63
  2.15.3 Typical NLP Research Methodology 63
  2.15.4 The R&D Potential of METAL 64
  2.15.5 Natural Language Processing using METAL 65
  2.15.6 Proposed Research using METAL 65
  2.15.7 References 66

2.16 Statistical Metaphors and the Computer Medium 68
2.16.1 Introduction 68
2.16.2 Dimensions of the Proposed Research 68
2.16.3 An Illustrative Metaphor: Actor-Based Fitting 69
2.16.4 References 69

3. Education 70
3.1 Artificial Intelligence Education at Texas 70
3.2 Courses in Artificial Intelligence 70
  3.2.1 Computer Sciences 70
  3.2.2 Electrical Engineering 72
  3.2.3 Mechanical Engineering 72
  3.2.4 Philosophy 72
  3.2.5 Linguistics 73
  3.2.6 Psychology 73
3.3 Fellowships 73
3.4 Education for Army Personnel 74
  3.4.1 Annual Conference 74
  3.4.2 Graduate Training for Army Personnel 74
  3.4.3 Summer Short Courses 74
  3.4.4 Summer Institutes 75
  3.4.5 On-Site Courses and Consulting 75

4. Management 77
5. Equipment 78

6. Budget 79
  6.1 Overall Budget 80
  6.2 Budgets by Projects 81
  6.3 Extensions to GLISP 84
  6.4 Physics Problem Solving 84
  6.5 Automatic Theorem Proving 85
  6.6 Device Descriptions in Text Knowledge Systems 85
  6.7 Target Recognition and Tracking 86
  6.8 Intelligent Interfaces, Problem Solving Under Uncertainty 86
  6.9 Program Transformation for Parallel Execution 87
  6.10 Statistical Metaphors and the Computer Medium 87
  6.11 Integrating Knowledge into Problem Solving Search 88
  6.12 Parallel Algorithms and Architectures for Problem Solving 89
  6.13 Discourse Representation and Processing 90
  6.14 Automated Translation and Natural Language Processing 91
  6.15 A.I. and Robotic Manipulation 92
  6.16 A.I. Project Joint Resources 93

I. Vitae 94

II. University Information 95
1. Introduction

We are pleased to submit to the Army Research Office this proposal for establishment of an Artificial Intelligence Project within the Department of Computer Sciences at the University of Texas at Austin. We believe that ARO's initiative presents a unique opportunity for both the University and the Army. Establishment of the Artificial Intelligence Project will allow the University of Texas rapidly to become one of the world's foremost institutions in Artificial Intelligence (AI). Establishing this Project at the University of Texas (UT) will allow the Army to meet its goals of increasing AI research and education by expanding what is probably the best and most broadly-based AI program outside the top three AI universities (Stanford, Carnegie-Mellon, and M.I.T.).

In this introductory section, we discuss the University; the strengths in AI that already exist here, and the equipment base that is available for research. As part of this proposal, we invite the Army to send a team to the University of Texas for a site visit so that we may demonstrate what the University has to offer as a site for an ARO-sponsored Artificial Intelligence Project.

1.1 The University of Texas at Austin

The University of Texas at Austin is currently celebrating its Centennial year. With a student body of approximately 47,000, U.T. Austin is one of the largest State universities, and one of the highest in quality. In its Centennial year, the President of the University, Dr. Peter Flawn, has established the goal of making the University one of the world's leading research universities by the end of the decade. The University's lofty goals, supported by its large endowment, are described in a recent article in Science (see Appendix II).

The Department of Computer Sciences at U.T. is already recognized as one of the nation's best; the Department was rated among the top ten nationwide in the recent Jones-Lindzey report on graduate programs in the sciences. The Department currently serves over 1500 undergraduate computer science majors, about 120 master's students, and about 100 Ph.D. students; there are 33 full-time-equivalent faculty. Last year, the Department, in conjunction with the Institute for Computer Science (located on campus) was awarded a five year NSF CER grant to support major experimental work in computer science.

The excellence of the Department of Computer Sciences was a major factor in the decision of the Microelectronics and Computer Technology Corporation (MCC) to locate its facilities in Austin. In conjunction with MCC's decision, the University has made a major commitment to expand its computer research capabilities. Fifteen faculty positions, including several endowed chairs and professorships, will be added to the Department over the next three years. The University will provide $2.5 million in
research equipment, $500,000 per year in research support, and $375,000 per year in graduate fellowships. A similar commitment has been made for strengthening computer engineering research in the Electrical Engineering Department. These commitments will allow the University rapidly to become one of the best in the world in Computer Science.

The Linguistics Department at U.T. is well known, with world-renowned experts in formal syntax and semantics and in computational linguistics. The Linguistics Department is ranked in the top three nationwide.

1.2 Artificial Intelligence at U.T. Austin

Artificial Intelligence has been an area of strength at the University since the founding of the Department of Computer Sciences. Woodrow W. Bledsoe, Ashbel Smith Professor of Mathematics and Computer Sciences, has been a pioneer in the field of Automatic Theorem Proving; Professor Bledsoe is currently President-Elect of the American Association for Artificial Intelligence, whose next annual conference (August 6-10, 1984) is to be held in Austin. There are nine faculty in the Department whose research is in Artificial Intelligence, and eleven faculty and senior researchers in other departments whose work is closely related to AI. Two new books on AI have been written by our faculty: Artificial Intelligence by Elaine Rich (published this year by McGraw-Hill) and Computations from the English by Robert Simmons (published this year by Prentice-Hall). The Cognitive Science Center at U.T. conducts research in areas of psychology, linguistics, and philosophy related to AI. The Linguistics Research Center has produced an operational and highly successful machine translation system (German to English, with other languages being added) with major funding by the Siemens corporation of West Germany. The combination of U.T.'s many areas of strength in AI provides a strong base from which U.T., with funding from ARO, can rapidly become one of the world's leading institutions in AI research and education.

1.3 Areas of Strength in Artificial Intelligence

In order to convey the breadth and depth of Artificial Intelligence research at the University of Texas, the paragraphs below describe our existing work in each area.

1.3.1 Problem Solving

The University of Texas has a number of research efforts involving automatic or human-machine problem solving. These include physics problem solving, proving correctness of programs, proving theorems in mathematics, theoretical studies of search procedures, and reasoning with uncertainty.
1.3.1.1 ISAAC Program

The ISAAC program [Novak76, Novak77, Novak80] is a program that can read, understand, and solve physics problems stated in English; the problems solved are in the domain of rigid body statics. The program reads a problem statement (for example, a textbook physics problem) stated in English; as it does so, it builds a model of the objects in the problem and their relationships. Once the problem has been read, the program constructs a geometric model that relates the objects in the problem to a common coordinate system. The program determines the role that each object plays in the problem, based on the type of the object and on the physical context in which it occurs; for example, a man might be modeled as a weight if he is standing on a ladder, or as a pivot if he is carrying a ladder. Equations are written describing the relationships of the objects according to physical laws; these equations are solved symbolically, using a small symbolic algebra system, to produce the desired answer. Finally, the program constructs a picture model for the problem and draws a diagram of it on a graphics terminal. The ISAAC program, running on a low-cost desktop computer, reads English at a speed of 5,000 words per minute; it solves a complete problem in two to five seconds. The program is significant both in its ability to understand multi-sentence connected discourse and in its ability to create a formal model, sufficient for problem solving, from an informal and incomplete English description of a problem.

1.3.1.2 Problem Solving Search Algorithms

Problem solving search algorithms have found wide use in many areas of computer science and engineering. Applications abound in such diverse areas as game playing, decision analysis, pattern recognition, robotics and theorem proving. Because of the somewhat differing requirements of the various problems in which search algorithms are used, different search procedures have evolved. For example, Alpha-Beta, SSS*, and B* for minimax search; A* and other heuristic search procedures for state space search; AO* for problem reduction search; and various Branch-and-Bound and Dynamic Programming procedures for combinatorial optimization problems. Although many of these procedures have been thought to be related to each other, considerable confusion has existed regarding their true nature and their interrelationships.

Kumar has developed a methodology whereby most of these procedures can be viewed in a unified manner. It reveals the true nature of these procedures, and clarifies their relationships to each other. The methodology also aids in synthesizing (by way of analogy, suggestions, etc.) new variations as well as generalizations and parallel implementations of the existing search procedures. Currently Kumar is working on the extensions of this methodology. The aim is to develop a methodology for integrating problem specific knowledge into general problem solving search procedures. This research is likely to aid in building theoretical foundations of expert problem solving systems.
1.3.1.3 Parallel Algorithms

Our unified approach to problem solving search procedures has helped synthesize parallel implementations of a number of search procedures. In particular, Kumar has developed two general schemes for performing Branch-and-Bound (B&B) search in parallel. These schemes are applicable in principle to any problem that can be solved by the B&B technique. But the effectiveness of a parallel implementation is expected to be dependent upon the nature of the specific problem being modeled. We have used these schemes to implement parallel formulations of SSS*, a versatile algorithm having applications in game tree search, structural pattern analysis and And/Or tree search. We have investigated the performance of parallel SSS* in the context of game tree search. A number of parallel game tree search procedures have been proposed and implemented. Simulation results show that Kumar's parallel implementations of SSS* perform quite well in comparison to these others.

1.3.1.4 Architectures for Problem Solving Search

Kumar is currently investigating the effectiveness of these parallel formulations in a variety of domains. We are also interested in developing new special purpose as well as general purpose parallel architectures for performing problem solving tasks efficiently. This research will be greatly facilitated by the parallel processor TRAC, being built at the University of Texas. TRAC, the Texas Reconfigurable Array Computer, consists of 16 processing units connected to 81 memory modules via a dynamically reconfigurable banyan switch network. By changing the degree of the banyan switch, TRAC type systems can be built to provide any desired degree of interprocessor communication. Furthermore, TRAC can dynamically reconfigure its processor, memory and I/O resources to implement a variety of parallel computation architectures, which makes it a very convenient environment for testing the performance of parallel algorithms as well as for research in new parallel architectures for AI problem solving tasks.

1.3.1.5 Problem Solving under Uncertainty

Artificial intelligence programs must be able to reason even when they possess only incomplete or uncertain information. A variety of techniques for doing this have been proposed. Such reasoning must explicitly manipulate measures of completeness and certainty in order to be able to ascribe such measures to final conclusions. Our previous work in this area has attempted to show how such explicit certainty reasoning can solve some traditional problems in common sense problem solving. We are continuing this work in an attempt to solve additional reasoning problems and with the intent of using measures of certainty as heuristic guides to problem-solving search processes.
1.3.2 Software Tools for A.I.

The University of Texas has a tradition of developing the software tools necessary to produce working A.I. systems. For example, the Lisp dialect used on large computers of Control Data Corporation, UTLISP, was written at the University of Texas in the late 1960's. Our tools have often been exported to other sites for use in their A.I. work. Several current software tools will be described: GLISP, HCPRVR, and two theorem provers.

1.3.2.1 GLISP

GLISP is a high-level language with abstract data types that is compiled into Lisp. It allows the programmer to describe the objects with which a program deals in a single place; programs can then be written in terms of objects and operations on objects rather than in terms of the implementations of the objects. The GLISP compiler converts the programs written in terms of objects into efficient programs in terms of the implementations. This allows application programs to be shorter (by a factor of two to three), easier to understand, easier to modify, and easier to transport. Associated with GLISP is a data inspector, called GEV, that interprets data according to abstract-data-type specifications. GEV allows the user to zoom in on features of interest, to examine computed properties of objects, and to write programs interactively by menu selection. The automatic programming feature allows a non-programmer to write and run a program comprising a page of Lisp code in a few seconds.

The GLISP compiler is currently running, and is being used by both commercial and nonprofit laboratories for writing Artificial Intelligence and Expert System software. We believe that GLISP, with the extensions proposed later, will provide real power for rapidly writing correct and efficient software.

1.3.2.2 HCPRVR

HCPRVR is a Horn Clause Theorem Prover programmed by Dan Chester to realize PROLOG semantics in LISP syntax. It is notable for its brevity and convenience of use in that it compiles into 1800 words of binary storage in LISP and thus augments the embedding language by introducing a basic theorem prover for interpreting procedural logic programs. It is available on several of our computers, and has been used for teaching procedural logic programming, for experimenting with robot block stacking problems, for constructing natural language interfaces to a document database, for applying English grammars, for translating small subsets of English and Japanese, for applying schemas and scripts to text, for developing experimental expert systems, and as a major basis for the Text Knowledge System (a natural language question answerer for a fifty page text).

HCPRVR incorporates a number of user utilities: for tracing and debugging, for asserting, marking, deleting and saving sets of axioms, for random selections, and for using LISP functions as especially powerful pseudo-axioms. It is available in several
dialects of LISP and has been used in about a dozen laboratories around the country. It serves primarily as a convenient design language to experiment with initial versions of AI systems; when the design is successfully established in HCPVR, it is often desirable to translate the resulting system to LISP and compile it at the assembly language level.

1.3.2.3 Theorem Provers

As will be discussed in some length below, The University of Texas has long been a center of research in automatic theorem proving. Over the years, many theorem-proving systems have been developed. Two of these are in use now as tools for other research efforts. These include one of Bledsoe's provers, currently being used in program verification work, and the induction-based prover of Boyer and Moore that is used for verification and also for work in programming methodology.

1.3.3 Mechanical Theorem Proving

In mechanical theorem proving, The University of Texas is one of the world leaders, for several reasons. We have a particularly large number of exceptionally successful researchers who come from both computer science and mathematics. Those researchers are among the few who have not only proposed new ideas for theorem provers, but have programmed them and exercised them extensively on a number of difficult theorems and program verification applications.

The present group of researchers in mechanical theorem proving at The University of Texas includes:

- W.W. Bledsoe, Ashbel Smith Professor of Mathematics and Computer Sciences, has been a leader of the automated theorem proving community since its birth and has generated a number of exceptional Ph.D. graduates. He recently became the President-Elect of the American Association for Artificial Intelligence (AAAI). Bledsoe has produced classic results in many areas of mechanical theorem proving, e.g., [1, 2, 3, 6]. He has been particularly concerned with mechanizing proofs as they might be performed by a human (natural deduction).

- R. S. Boyer and J S. Moore, of the Department of Computer Sciences and the Institute for Computing Science, have won world-wide acclaim for their research on the mechanical treatment of mathematical induction [4]. They recently were awarded the McCarthy Prize for Program Verification. Boyer and Moore's functional logic is particularly suitable for program verification and has, among other things, been used for an implementation of a verification condition generator for FORTRAN programs [5].

- D. I. Good heads the University's Institute for Computing Science. He and his group have developed a language, Gypsy, for verifiable parallel programs [7]. Good's group recently succeeded in a mechanical proof of a Gypsy program of more than 4000 lines of code [8].
• C. Lengauer, of the Department of Computer Sciences, has a background in formal semantics and verification and is working on the automation of a programming methodology, i.e., a formal calculus for the derivation of programs [9]. He is using the Boyer-Moore prover.

• R. F. Simmons, of the Department of Computer Sciences, is world-renowned in the field of natural language understanding. He uses a procedural logic, HCPRVR (see 1.3.2.2), developed by Dan Chester and himself to check the consistency of English sentences with respect to a set of text axioms [10].

1.3.3.1 References


1.3.4 Natural Language Processing

Natural language studies at the University of Texas have a history stretching back to the early 1960s when Robert Lindsay constructed a first prototype for answering questions about family trees, and Roger Elliott built a general inference system for dealing with natural English notions of the transitivity of directions and adjacencies. When Robert Simmons joined the CS Department in 1968, a period of intensive research ensued that resulted in the excellent work of B. Bruce on the logic of natural language case systems, the mathematical theory of Semantic Networks by Gary Hendrix, the pictorialization and solution of natural language physics problems by Gordon Novak, several approaches to the computation of text structures by M. Smith, R. Young, and R. Alterman, comparative studies of the effectiveness of natural language parsers by Jonathan Slocum, machine learning of case grammars by H. Smith, and detailed studies of semantic structures in English dictionaries by R. Amsler. This year Simmons' book, Computations from the English, describes a highly practical approach to natural language parsing, question-answering, paraphrase, and robot command systems using a procedural logic system in LISP -- all based on the fifteen years of research and the score of dissertations in natural language research during that period. Slocum's "Machine Translation: its history, current status, and future prospects," a keynote address to COLING, the international computational linguistics society, appears to be a landmark paper in the area; his METAL system has produced good translations from German to English for over a thousand pages of text.

A broad range of courses in linguistics, philosophy of language, psycholinguistics, computational linguistics, formal language theory, artificial intelligence, and expert knowledge systems has been developed over this period, and interested graduate students can be thoroughly trained in all aspects of the area. With increased funding of natural language research, many more students can be rewarded with research assistantships for following their interests.

1.3.5 Cognitive Science

The University of Texas has several research centers engaged in research on, or relating to, A.I. One of these centers is the Center for Cognitive Science, which engages the talents of many linguists, psychologists, and philosophers. Among other things, personnel of this center are investigating natural language processing, and one group is currently developing Discourse Representation Theory, described in detail in the Research Section. Primary personnel for this project will include Hans Kamp (Philosophy), Carlota Smith (Linguistics), Daniel Bonevac (Philosophy) and Nicholas Asher (Philosophy). Hans Kamp is the founder of discourse representation theory, and is recognized as one of the foremost philosophers in the world currently working in logic, semantics and the philosophy of language. Smith has published extensively on the syntax and semantics of natural language. Bonevac has published articles on the philosophy of mathematics, logic, and formal semantics; his book, Reduction in the
Abstract Sciences, won the Johnsonian Prize in Philosophy in 1980. Asher has worked on problems in language processing and linguistic understanding and has published papers on natural language semantics.

Several other individuals will consult on this project, engage in related research, and participate in the A.I. educational program. They include: Lauri Karttunen (Linguistics), Robert Wall (Linguistics), Robert Boyer (Computer Science) and Robert Causey (Philosophy). Karttunen has done major work in the formal semantics of discourse, as well as in other areas of semantics, and is responsible for a computational implementation of the discourse representation construction algorithm. Wall, coauthor of Introduction to Montague Semantics and author of An Introduction to Mathematical Linguistics, works on syntax, formal semantics and computational linguistics. Causey, author of Unity of Science, works on scientific methodology and applications of computer technology in education and educational administration.

1.3.6 Automated Translation

The Linguistics Research Center of the University of Texas is actively engaged in the development of an Automated Translation (AT) system, funded entirely by the German multi-national firm Siemens AG. Employing state-of-the-art Natural Language Processing techniques, the LRC has constructed a translation system (METAL) widely recognized to be at the cutting edge of AT technology. A German-English AT system is nearing production application at Siemens. An English-German AT project is well underway, and projects for German-Spanish, English-Spanish, and English-Chinese are getting started. The METAL system has already been demonstrated to be cost-effective (i.e., cheaper as well as faster than human translation, all costs included), and Siemens has announced its intention to add other languages in the near future. The METAL system is intended for use in an environment of technical texts (especially operation and maintenance manuals), and human revision of the output is expected -- as is the case with human translation in most places around the world. (I.e., where translation is taken seriously, all translations are edited, whether performed by humans or computers.)

Essentially all modern applications of AT systems are in an environment of technical translation followed by human revision. The usual situation is that an AT system will produce translations far faster and cheaper than any human translator could (human translation rates for technical material lie in the neighborhood of 4-6 pages/day), but revision of AT output proceeds more slowly than is the case with revision of human translations (which averages 8-12 pages/day). What distinguishes the METAL system from other AT systems (aside from the fact that it is based on state-of-the-art as opposed to decades-old technology) is that, in recent experiments conducted by the project sponsor, the revision rate of technical texts translated by METAL has actually exceeded the revision rate [by the same editors] for like texts translated by humans (15-17 vs. 10-12 pages/day). Thus, the total cost for technical translation using METAL has been shown to be very much less than that for human translation of the same kind of material
-- for both the initial translation and the revision. AT at the Linguistics Research Center of the University of Texas is truly coming of age.

Since the METAL system is built upon general state-of-the-art Natural Language Processing technology (highly efficient parsing and generating programs, large grammars and dictionaries of multiple natural languages, etc.), and has proven itself cost-effective in a practical, large-scale Natural Language Processing application (AT), the Linguistics Research Center has demonstrated an almost unparalleled ability to engage in productive R&D in applied Natural Language Processing. In order to assure continued success in the future, we propose herein to engage in long-term research using METAL as a central tool for both initial inquiry (gathering data) and testing resulting theories of natural language analysis, translation, and synthesis.

1.3.7 Intelligent Interfaces

As the use of computers spreads into more and more areas of human endeavor, it becomes increasingly important that computer systems be easy for people to use. There are some simple ways in which ease of use can be increased, such as providing clearer error messages and using menus to display alternatives. But the best way to increase the usability of computer systems is to make the systems themselves more powerful by augmenting their knowledge and problem-solving abilities. Artificial Intelligence techniques provide ways of doing this. At the same time, since the problem domains in which intelligent interfaces are important are so varied in nature, work in this fundamentally applied area can suggest new techniques that can increase our understanding of basic knowledge-representation and problem-solving issues.

Our previous and current work on this problem spans several areas. One is the use of models of individual users to enable a program to tailor its performance to the specific needs of each of its users. Another is the design of an on-line help system that answers questions by reasoning about the system for which information is desired and constructing an appropriate reply to each question. Yet another is the design of an operating system interface that corrects user errors and offers useful advice based on its knowledge of standard user goals.

1.3.8 Laboratory for Image and Signal Analysis

The Laboratory for Image and Signal Analysis, established in 1980 under the direction of J.K. Aggarwal of the Department of Electrical Engineering, is devoted to the development of computational models and techniques for the analysis of one-dimensional and two-dimensional signals for application in the fields of signal processing and image analysis. Part of the Bureau of Engineering Research, the Laboratory is composed of faculty drawn from the Departments of Electrical Engineering, Computer Sciences, and Mechanical Engineering. The laboratory has in its possession state-of-the-art equipment in high-resolution T.V. cameras, laser ranging devices, digitizers, and high-speed
interfaces to the computing facilities. These latter facilities include two VAX 11/780 high performance computers coupled through a 70M bit/sec bus. The system allows for expansion to include up to 16 VAXes. Attached to these systems are a variety of high-performance peripherals which include color graphics terminals, 3-D graphics and color gray scale hard copy devices. Current research includes analysis, synthesis, implementation and evaluation of two-dimensional digital filters, time-varying digital filters and application of digital signal processing techniques to seismic data processing; and development of models for the formulation and analysis of parallel image processing algorithms and techniques for determining suitable parallel computer architectures for high-speed image analysis.

The results of the Laboratory's research are presented at national and international conferences and published in laboratory reports and scientific journals. Reprints of recent reports and Journal publications are available from the Laboratory on request.

1.3.9 Robotics

Research in robotics at the University of Texas is centered at the Flexible Automation/Robotics teaching and research laboratory housed within the Department of Mechanical Engineering (ME). The primary goal of the robotics research laboratory is to develop techniques and approaches that will allow for increased use of robotic manipulators in the manufacturing environment. The robotics teaching laboratory is fairly well-equipped due to its being part of both the Manufacturing Systems Graduate Engineering Program and the Mechanical Systems undergraduate and graduate engineering program. Specifics concerning the equipment are discussed in another section of the proposal.

The research activities conducted within the laboratory are in three general areas: offline programming or task decomposition, intelligent feedback control, and the development of models and system design techniques. The major aspects of each of these projects are discussed below. In each case, some of the completed projects, current projects, and anticipated future projects are discussed.

The research in offline programming of robots represents a bottom-up approach. That is, the first few projects tend to be related to hardware control, and the latter and more current projects deal with intelligent control. Thus, the lower level projects are computationally intensive, while the higher level and latter projects deal with inferences. Because of the unique nature of robotic devices, both types of problems need to be solved in a dependent fashion. The following projects represent steps within the hierarchical control of robotic manipulators. The projects involve:

1. Modeling the kinematics of robot linkages [Colson, Perreira, 1983].
3. Developing pose measurement transducers (6 degrees of freedom).

4. Developing statistical measurement techniques to determine the actual manipulator's performance [Perreira, Colson, 1983].

5. Developing analytical techniques based on the above measurement results to determine the errors made by using nominal linkage parameter data, servo-controller errors and measurement system errors on system performance, and to estimate the system parameter values that cause them. [Perreira, Ibarra, work in progress].

6. Developing analytical techniques which use the estimates of the actual linkage parameters, servo and measurement system errors in determining the robot's actual "footprint". [Perreira, Smith, work in progress].

7. Developing a procedure to use the estimates of the system parameters and servo and measurement system errors to modify desired pose trajectories into a form that will allow the robot to achieve the desired trajectory within the confines of user specified performance. [Perreira, Smith, work in progress].

8. Developing intelligent procedures to modify the desired pose trajectory to incorporate differences between the actual environment and a model-predicted environment as determined and made available from both Computer Aided Design and Computer Aided Manufacturing system programs and through external sensory information. In our case, a vision system is used to supply this information. [Perreira, Mehta, work in progress].

9. Developing intelligent procedures to modify the desired assembly task decomposition and order (within the confines of achievable trajectories) because of unpredicted changes in the assembly work cell.

A second broad-based project concerns the use of intelligent feedback control of robotic manipulators. The project includes:


2. Development of a recursive set of Hamilton canonical equations for the actuation of link motion. [Perreira, Nanayakarra, work in progress].

3. Development of a multiprocessor system to implement the above procedure for dynamic control of a manipulator using a parallel computational hierarchy. [Chapman, Perreira, 1983, and work in progress].

4. Development of a parallel hierarchical dynamic controller for a robotic manipulator where obstacle avoidance and other environmental conditions require quick intelligent modifications of the control processors.
The third project is composed of a number of pure analysis and design projects for manufacturing environments. The projects include:

1. Analysis, design, and testing of automatic feedtrays for parts presentation. [Perreira, work in progress].

2. Analysis and design of robotic manipulators based on work volume, dexterity, actuation, and usage requirement. [Perreira, work in progress].

The three major project areas, task decomposition, feedback processing, and modeling, represent various levels of a complete intelligent hierarchical control system for robotic manipulators. A proposal specific to the development of such a hierarchy is described in detail in the research section of the proposal.

1.4 Research Computer Facilities

1.4.1 Computer Sciences Facilities

The University of Texas already has a strong base of computer facilities needed for AI research. A DEC-2060 is located within the Department of Computer Sciences and is dedicated to research in AI and related areas (theorem proving, machine translation, cognitive sciences). The Department also has a VAX-11/780 for research; these machines are connected to ARPANET and CSNET. Research groups on campus have approximately 12 Lisp machines of various types; these are connected to the DEC-2060 by Ethernet.

1.4.2 Flexible Automation/Robotics Laboratory

The Flexible Automation/Robotics Laboratory currently contains the following equipment:

- IBM 7510 Robotic Manipulator (RS/1)
  A box-frame, closed-loop controlled, hydraulically actuated, six degrees of freedom robot system controlled by a Series I minicomputer using fairly sophisticated software known as A Manufacturing Language (AML). The robot has been retrofitted with the 7565 (RS/2) software, the latest AML software.

- IBM White Cloud
  A developmental robotic system used in designing the RS/1 and RS/2. It includes its own Series I minicomputer.

- Unimate PUMA 560
  An articulated, closed-loop, D.C. servomotor-actuated, six degrees of freedom robotic system controlled by an LSI-II minicomputer with six microprocessor joint controllers. The robot is one of the workhorses found in industry and is one of the most advanced systems available.
• Microbot Teachmover
  A five degrees of freedom, open-loop, controlled stepper motor activated robot, driven by a single microprocessor with a low level language stored in Eprom and interfaced to an Apple computer. The system is used by students in developing trajectory algorithms.

1.4.3 University Facilities

The University has several computer facilities that are available for research use as needed. These include dual CDC CYBER 170/750 computers, an IBM 3081, a DEC-2060, an Advanced Graphics Laboratory with facilities for machine vision work, and numerous VAX computers. The Computer Science Department's machines are connected to the high-speed network that interconnects all the campus computers. A MICOM port contention system switches 1300 terminal lines among 600 computer ports.

As this brief overview shows, the Department and the University already possess a solid substrate of computing equipment needed for AI research. By adding incremental amounts of specialized equipment, such as Lisp machines, a computing environment second to none can be created for AI research and education. In effect, the Army's funds will be leveraged by the substantial amounts of equipment already in place.
2. Research

2.1 Introduction

One of our major goals in submitting this proposal is to obtain funding to increase the quality and quantity of AI research performed at the University of Texas. Additional funding will aid our research efforts in several ways:

1. Having first-rate research equipment (in particular, Lisp machines) will allow our work to proceed faster and will eliminate constraints now imposed by limited availability of machine cycles.

2. Faculty are allowed to *buy* their time using grant funds, reducing their formal teaching duties by up to one-half. This allows them to devote more time to research.

3. Having first-class equipment and reduced teaching loads will allow us to attract excellent AI faculty to fill some of the 15 Computer Science faculty positions to be added over the next three years. We expect to integrate new AI faculty into the research programs proposed here.

4. The funds requested in this proposal will allow us to hire postdoctoral researchers into an environment that will be competitive with those found in industry.

5. Equipment, fellowships and research-assistant funding will allow us to attract and retain the best students for AI research. We have seen far too many good students leave AI and go into other areas where more financial support has been available.

In all of these ways, ARO funding will increase the quality and quantity of AI research by allowing more people to perform research on better equipment.

2.2 Research Structure

One of the strengths of AI research at the University of Texas is the breadth of coverage of areas of AI. We propose to structure our research program to take advantage of this diversity, while maintaining communication among the researchers and adequate management controls. Each research project will be managed by one or more Principal Investigators, under general direction of the Project Director and Executive Committee. Communication will be fostered by an organized seminar series at which investigators will present lectures on their research.

Each of the proposed research projects is described in the following sections. We propose that the initial commitment of research support from the Army be made for five years. The various proposed research projects have varying durations. We expect that
some research projects will be completed prior to the end of the five-year period, that new research projects will be started as new AI faculty are added, and that some projects will change direction as new research opportunities present themselves. In order to provide flexibility in the research program, we propose that the Army provide the amount of research funding that we request for the first year, and similar amounts (adjusted for inflation) in succeeding years. Prior to the start of each fiscal year, we will submit to the Army a revised proposal for research projects and budgets for the next year; these will be subject to review and acceptance by the Army. This method will provide flexibility for the research program together with adequate control over the program's direction.

In the following sections, the individual research projects that are proposed are described in detail.
2.3 Extensions to GLISP

Principal Investigator: Gordon S. Novak Jr.

2.3.1 Introduction

GLISP is a high-level language with abstract data types that is compiled into Lisp. It allows the programmer to describe the objects with which a program deals in a single place; programs can then be written in terms of objects and operations on objects rather than in terms of the implementations of the objects. The associated GEV data inspector allows programs to be written automatically based on selections from menus derived from object descriptions.

2.3.2 Data Description Language

GLISP contains a data description language that is sufficient for describing most data structures used in Lisp programs. We wish to extend this into a general data description language that is capable of describing any data structure from the CPU instruction and data format level to very high level data structures. Such a data description language will make it possible to use the GLISP compiler for a wide variety of applications and for experiments with abstract algorithms, as described below.

2.3.3 Abstract Algorithms

It is unfortunately the case that little of the knowledge of algorithms that has been accumulated is available in the form of program libraries. Those library subroutines that are available generally deal with very simple data structures (e.g., FORTRAN arrays). For other well-known algorithms (e.g., how to traverse a tree in preorder), the variation in possible representations of the data structures (in this case, trees) is so wide that there is little chance that any library subroutine would work for the particular data structures of interest at any given time. An additional problem is that it is often necessary to combine algorithms, e.g., to combine traversal of a tree with certain operations on the nodes; traditional subroutine calls do not permit such combination. As a result of these problems, much of the effort of programmers is spent in customizing well-known algorithms for particular applications.

The GLISP compiler is able to take an abstract algorithm, written in terms of abstract data types, and specialize it into an executable program for particular data types that are instances of the abstract types. For example, a general program for drawing diagrams of tree structures can be specialized into a program to draw a particular type of tree in a particular way. However, the representation and specialization techniques that are now implemented are not sufficiently robust for general use. More research is needed on ways to represent features of abstract types and constraints on those features, ways to view the same data from different perspectives during the compilation process, and semi-automatic ways to unify instance types with abstract types. Our goal is to
produce a system that can *negotiate* an interface between the user's data types and those expected by library programs, then produce versions of the library programs that will work with the user's data. We believe that most of this negotiation can be done by menu selection in a fashion similar to that used in our automatic programming system; such techniques could achieve the vital goal of interchangeable parts in the domain of software.

2.3.4 References


2.4 Expert Problem Solving in Physics

Principal Investigator: Gordon S. Novak Jr.

2.4.1 Introduction

We propose to investigate methods by which a computer program can understand an informal statement of a physics problem, expressed in English and diagrams, and can use this understanding to appropriately model the physical situation embodied by the problem and solve it. We will focus on the domain of elementary physics problems, since physics is the best-formalized technical domain, yet is complex enough to be challenging. We will build on previous research [Novak76, Novak77, Novak80] on the ISAAC program, which is able to understand rigid body statics problems stated in English, solve the problems, and draw diagrams of them. We propose to extend the program to cover a substantial range of physics problems; this will involve basic research in man-machine communication, representation of knowledge, and control of the problem solving process.

2.4.2 Man-Machine Communication

Most computer programs require that their input be stated in a rigid, prespecified format; this makes it difficult to describe complex problems. On the other hand, human communication using natural language and diagrams is informal, brief, and very flexible. In order for a computer program to understand such forms of communication, it must have the knowledge and procedures to enable it to infer the appropriate context and likely consequences from an abbreviated description. Physics provides an excellent domain in which to investigate computer understanding of informal descriptions, since the result of the understanding process is strongly constrained: it must contain the information necessary for understanding intersentential references and for actually solving the problem. We propose to expand ISAAC's natural language abilities as needed to expand the area of physics covered, and to begin to investigate the use of text and diagrams together for problem specification.

2.4.3 Representation of Knowledge

While computer programs usually represent objects from only a single, limited point of view, a robust problem solver requires the ability to view objects from many perspectives. For example, a real-world object such as a car may be viewed as a point mass in one problem, as a rigid body in another, or as an energy-conversion machine in a third. Such representations are metaphorical in the sense that they highlight approximations of certain characteristics of objects while ignoring other characteristics. The "canonical objects" of elementary physics (e.g., "point mass" and "rigid body") are well understood, making physics a good area in which to investigate the creation of formal models from informal descriptions. We propose to develop formal representations for the canonical objects of elementary physics and for the conceptual frameworks (e.g., coordinate systems) needed for solving physics problems.
2.4.4 Control of the Problem-Solving Process

While the expert problem solver is able to model a given real-world object in many different ways, only a few of the many possible object models will be used in solving a particular problem. The choice of physical models for a real-world object or situation is determined by the context (explicit and implicit) in which the object occurs as well as by the characteristics of the object itself; the ability to correctly "set up" a problem is a large part of problem solving expertise. The nature of this expertise is poorly understood in formal terms even by skilled problem solvers. We believe that this expertise can be modelled by rules which diagnose the type of a problem and prescribe appropriate methods for solving it; we propose to develop an expert system containing a set of rules that is sufficient to cover the subject matter of first-semester college physics.

2.4.5 Applications

Two important classes of applications can follow from the proposed research. The first is development of expert consultant programs that can be used to solve problems in technical areas. Because such programs could communicate informally in natural language, it would not be necessary for the user to be a computer expert or to enter a complex formal description of a problem in order to use such a system. A second potential application is in teaching programs. Problem solving is both difficult to learn and difficult to teach, in part because the human expert has "compiled" the rules for problem solving in a form no longer accessible to introspection. Once the rules for problem solving have been made explicit, a program could be written to explain the problem-solving process and provide specific criticism of a student's problem solving.

2.4.6 Research Plan

Our initial focus will be development of representations and problem analysis rules for Newtonian mechanics. These will be tested by using them to solve a large number of textbook problems. The natural language understanding abilities of ISAAC will be extended to accept these problems and to produce the new representations as output. Diagrams of problems and protocols which track the problem-solving process will be produced on a high-resolution graphics display. Finally, we will expand the coverage of the program to wider classes of physics problems, and investigate the use of text and diagrams together for inputting problem specifications.

2.4.7 References


2.5 Device Descriptions in the Text Knowledge System

Principal Investigator: Robert F. Simmons

One of the more fascinating documents of our high-technology culture is the book, *How Things Work*. Technological devices ranging from the steam engine through atomic fission plants are briefly described in one page statements; on the facing page for each one is a schematic diagram showing how the device works. We propose, for two or three such devices to translate both the text and the diagrams to formal logic descriptions, then construct a dynamic, procedural model of the device, and relate the text and diagram descriptions to the model. The resulting structures will be tested for their ability to answer questions and solve problems concerning the device.

We propose, in addition, to develop an English lexicon and grammar to aid in the translation of the text and questions to logical representations.

The research will be conducted on a Lisp machine with windowed, bit-mapped displays. The system will be designed to illustrate the text by running the model of the device it describes, and to provide dynamic diagrams as part of the explanation in response to English questions and commands. We believe that the development of such a system will have great potential utility for natural language access to technical text databases, for training technicians, and for acting as a diagnostic consultant in the fashion of expert knowledge systems.

In our current research [Simmons 1983b] a text knowledge system (TKS) has been developed for fifty pages of *The Handbook of Artificial Intelligence* by Barr and Feigenbaum. The TKS (pronounced Tek) includes English dictionaries and grammar sufficient for questioning the Handbook sample, a procedural logic interpreter, and a text data management system for navigation throughout the texts it contains. On a LISP machine it can access up to 16 million words of memory using the operating system as its underlying paging mechanism. The system represents the Handbook text as a system of logical assertions, uses the grammar and dictionary to translate questions and commands into corresponding logical relations, then attempts to answer a question by proving it as a theorem with respect to the text assertions as axioms. Rules of inference are applied in this process both to paraphrase the question into the terms of the text and to combine disparate portions of the text that contribute to answering a single question. The system has so far been tested on a sample of about 100 English questions. While the natural language understanding capabilities are woefully weak by human standards, they are sufficient to deal with direct questions that are closely related to the text, and the data management system makes all parts of the text quickly and easily accessible to navigational commands given either in English or by keyboard or light-pen signals.

The TK System, by virtue of containing linguistic and logical apparatus and including a data management system, provides a sophisticated laboratory for the development of natural-language-based knowledge systems. As a result, the proposed research need not
be concerned with constructing the basic systems; it can concentrate almost entirely on the tasks of representation and question answering. The linguistic and logical interpreters are already there, the basic question-answering logic is present, and the only underlying systems that need be developed will be those for displaying and (perhaps) animating the figures.

2.5.1 References


2.6 Intelligent Interfaces

Principal Investigator: Elaine A. Rich

We propose to investigate the following questions, all of which are important in the construction of intelligent interfaces:

- How can one program be understood and explained by another? This is a key piece in the design of intelligent help [4] and advisory programs [1].

- How can models of individual users be inferred and used to generate behavior tailored to the needs of each specific user? [3,5].

- How can an intelligent tutoring system be designed so that the computer can serve as an instructor to people who must use it? [2].

To investigate these issues, we have three projects already underway. We propose to continue these projects and to expand them to explore issues that are now poorly understood. The next three sections contain outlines of these projects.

2.6.1 The Scribe Tutor

Traditional on-line help systems respond to user queries with canned pieces of text. They do no creative problem solving. Because of this, people often find that the only way they can actually get their problems solved is to ask questions of a human consultant. This human uses his/her knowledge of how the system of interest behaves to try to find a way that users can get their problems solved. Sometimes there is no solution to the problem as stated. Then the consultant will try to find a slightly perturbed problem that can be solved and ask the user if that solution is acceptable. We are working on the construction of an on-line help system that simulates this type of problem-solving behavior.

We are building a help system for the document-formatting program Scribe. This system will eventually consist of the following components:

- A natural language front end.

- A dialogue manager, which decides what the user meant, as opposed to what was literally said.

- A user modeler, which maintains a model of the knowledge and interests of each individual user.

- A question answerer, which takes a specific question in a precise question language and answers it using a knowledge base of information about Scribe.

- An answer constructor, which phrases the output of the question answerer in terms appropriate for the specific user.
• A natural language generator, which converts the answer into proper English.

A key issue in the design of this system is the way in which knowledge about Scribe’s behavior will be represented. We have chosen to represent that knowledge as code, augmented by as little else as possible. There are two major advantages of this approach:

• It means that the knowledge base never "bottoms out" in the sense that it does not possess enough detail to answer a specific question.

• It means that the knowledge base is guaranteed to be consistent with the Scribe system itself.

The work we have so far done on this project has focused on the question-answering component of the system and on the design of the programming language in which the knowledge for that system will be represented. We intend to pursue this work, with a two-pronged goal: to build a good help system and to increase our ability to reason about programs.

In addition, we intend to begin work on some of the other components described above. We are particularly interested in the development of the user modeler and in the dialogue maintainer and in the ways in which these two components interact with each other.

2.6.2 The Programming Tutor

Although there is ongoing research in the area of automatic programming, it will be necessary, at least for quite some time, for people to do a good deal of programming themselves. One way in which artificial intelligence techniques can be applied is to help people to learn to do this programming. We are working on a programming tutor that will be able to do just that. Besides being of practical interest, this effort is of theoretical interest because it requires an increased understanding of the structures used in programs and of the way that those structures relate to the domain-specific entities with which programs must deal.

We are working on a tutor for LISP. The tutor assigns to the student a problem to solve. It then examines the student’s program and compares it to programs that it can generate from a collection of abstract plans for solving the particular problem. It attempts to make a correct program match the student’s program as closely as possible. It then looks at remaining differences and isolates bugs in the student’s program. The goal then is to discover what underlying misconceptions caused the student to produce those bugs. Doing this requires knowledge both of programming concepts and how those concepts can be misunderstood, as well as of a model of the individual student that suggests likely misconceptions. Thus this project, as well as the Scribe help system, rely heavily on both an understanding of the structure of programs as well as on the ability to build and exploit models of individual users of the respective systems.
2.6.3 The Intelligent Operating System Interface

Regardless of how many capabilities can be transferred from a machine's users to the machine itself, people will have to interact with machines, if only to state their problems. Traditional operating-system-level interfaces accept commands and execute them, but they know nothing about the user's overall goal. Because of this, they cannot interpret incorrect commands properly, nor can they give the user any advice. This is analogous to the situation just outlined with respect to Scribe. In that domain, we have increased the system's problem-solving ability by constructing a help facility that can answer detailed questions about Scribe's behavior. In the operating system domain, we are exploring a different approach; we are modifying the interface itself so that a user's command is interpreted by a system that has both a long-term model of that person as well as an estimate of the user's current goal. Using these models, errors can be corrected, advice can be given, and feedback can be provided to the system itself so that performance can be tailored to the exact demands of current users.

We are exploring this approach using the TOPS20 operating system on a DEC 2060. This system is attractive because it provides a wide array of capabilities with relatively little unnecessary hassle to the user. The system is already "friendly" in most of the usual senses, so performance improvements must be based primarily on knowledge rather than simply on superficial syntactic changes.

We have modified the existing operating system interface so that we can collect data on the kinds of errors people actually make. In the next phase of the project, we will use those data to design models of common user goals and ways of achieving those goals. We will then use those models to do two things:

- Improve the performance of the system from the user's perspective by making it more responsive to specific needs.
- Improve the performance of the system from the system's point of view by exploiting knowledge about how users will exploit the system in order to do such things as job scheduling in as close as possible to an optimal manner.

In this effort, as in the last two, the ability to model individual users, their needs, and their knowledge, will be very important.

2.6.4 References


2.7 Problem Solving under Uncertainty

Principal Investigator: Elaine A. Rich

Default reasoning is an important part of many problem-solving systems, since it provides a way of dealing with the lack of complete knowledge about the problem domain. Whenever default reasoning occurs, however, it is important to recognize that the truth value of the resulting proposition is different from that of a proposition that was derived directly from a set of definitely known facts. The truth values of any propositions derived from a proposition that is itself derived by default reasoning must also reflect the appropriate level of lack of certainty. We have begun to develop a theory of likelihood reasoning based on this idea [1].

Certainty factors provide help to a reasoning system in several ways, including:

- They provide a way of deciding on a single outcome even when there is conflicting evidence.
- They suggest paths that are most likely to lead to an eventual solution that is most likely to be correct.
- They make it possible to insert into a knowledge base relationships that are significant but that are sufficiently uncertain that, if the only truth values that were allowed were True and False, they would have to be omitted.

We intend to pursue this work in two ways:

- Continue the development from a theoretical point of view and look for classes of problems for which the approach seems well suited.
- Begin implementing the system so that it can be experimented with in the context of a problem-solving system operating in the context of a real knowledge base.

This work cuts across the boundary between knowledge representation and problem solving, since it is at the same time both a way to represent uncertain knowledge as well as a way to reason with it to derive both conclusions and measures of confidence in those conclusions.

2.7.1 References

2.8 Research and Development in Automated Theorem Proving

Principal Investigator: Woodrow W. Bledsoe

2.8.1 Introduction

Over the last several years our group at the University of Texas (the Bledsoe Group) has developed a number of computer programs which prove theorems in Mathematics and application areas (such as program verification). Some of these have been designed for interaction, whereby the user helps with the proof [14,15,13], while others were designed as "stand-alone" provers [1-2,8-10,18-23].

The more recent of the stand-alone provers, two of which are currently under development, use Agendas and what can loosely be called "higher-level goals" to guide their search. This approach seems to show promise and needs further development and testing.

2.8.2 Current Research

Larry Hines and Tie-cheng Wang, working independently on PhD dissertations, are developing multi-step heuristics to speed up proof discovery in Resolution-type provers, in particular for our inequality prover [2]. (These are expert systems). This prover [2] has had a good deal of success on certain theorems about general inequalities (not just ground inequalities) which arise from analysis; but it is only with the aid of these multi-step heuristics that we have been able to prove automatically such theorems as AM5, with a general purpose prover.

\[ \text{AM5. } a < b \land C1 \land C2 \land LUB1 \land LUB2 \land GLB1 \land GLB2 \]

\[ \text{--------- } \text{SOME m (} a < m < b \land \text{ ALL t (} a < t < b \implies f(m) < f(t) \text{) } \text{) ,} \]

where C1, C2, LUB1, etc, are as follows:

\[ \text{C1: } \text{ALL x in } [a,b] \land \text{ ALL } E > 0 \land \text{ SOME r < x \land ALL } s \land (r < s < x \implies f(x) < f(s) + E) \]

\[ \text{C2: } \text{ALL x in } [a,b] \land \text{ ALL } E > 0 \land \text{ SOME r < x \land ALL } s \land (r < s < x \implies f(x) < f(s) + E) \]

\[ \text{LUB: SOME L } \{ \]

\[ \text{LUB1: } \text{ALL x in } [a,b] \land \text{ ALL t (} a < t < x \implies f(x) < f(t) \text{) } \implies x < L \} \]

&

\[ \text{LUB2: SOME y (} y < L \implies \text{SOME z in } [a,b] \land \text{ ALL t (} a < t < z } \]
---> f(z) < f(t) & y < z < L )})

GLB ALL w in [a,b] SOME g {

GLB1: ALL x in [a,b] ( f(x) < f(w) ----> g < x )

&

GLB2: ALL y (g<y ----> SOME z in [a,b] ( f(z) < f(w) & g < z < y ) )}

AM5 is a first order theorem, derived from the higher order theorem:
A continuous function on a closed interval attains its minimum on that interval,
by instantiating (properly) the set-variable A in the least-upper-bound axiom: Every non-empty, bounded, set A of real numbers, has a least upper bound L; and similarly instantiating the greatest-lower-bound axiom.

There are of course many theorems similar to this, and more difficult ones, arising from intermediate analysis and analysis. Handling them automatically becomes a priority item if we are eventually to bring automation to mathematical proof search in analysis, in a useful way.

Our methods work equally well for theorems arising from application areas; indeed, some of our methods such as user-interaction and inequality provers [17,13-15] were designed for applications in Program Verification.

The prover [2] derives its effectiveness from three facts:

1. Ground Proofs are relatively easy;

2. VE (Variable Elimination) makes a clause "more ground";

3. Chaining on shielding terms tends to
   • make variables "eligible" for VE,
   • remove variables by instantiation.

These three facts led to the following strategy used in [2]:

1. If a ground refutation is possible, then do it.

2. If a variable can be eliminated by VE, then do it.

3. Try to make variables eligible for VE by chaining on shielding terms (see [2] for definitions and details).

4. Never chain on a variable.
These "higher-level goals" are a step in the right direction, but a large number of theorems, such as AM5 above, have several variables and shielding terms; they require expert care in choosing which variables are to be eliminated first, and which shielding terms are to be removed first to make those variables eligible for VE.

Hines is able to order the variables in a clause by a notion of "variable dependency" and attack them in the order of that dependency; similarly for shielding terms. He also employs a "counter-plans" mechanism for recovering from a failure. E.g., if a sequence of chainings and VE's has led to tautology or a dead end, then this mechanism proposes a new ordering (not just a permutation) of this sequence which will, if possible, eliminate the difficulty. He is developing an agenda mechanism to control all of these procedures.

Wang is developing a somewhat similar but altogether different set of heuristics, under control of an agenda mechanism, to speed up resolution proofs in general, and the inequality prover [2] in particular. He has already had a great deal of success, including the proof of AM5 and others like it.

These provers are "general purpose" and are essentially complete for first order logic (some versions are complete); some utilize algebraic simplification, rewrite rules, and subroutines specifically designed for inequalities, equalities, set-variable instantiation [21], and some generate and use examples to help in the proof search.

We propose to continue to develop these provers and others like them that incorporate our new ideas (and those of researchers from other research groups), and to exercise them on a number of graded, difficult theorems.

Part of this effort would involve careful documentation so that they can be more easily exported to other researchers. This would include user instructions and a "help" facility, as well as a graded list of documented examples. However, the main reason for documenting the provers and exercising them on hard examples is to help see how their basic strengths can be enhanced.

2.8.3 References


2.9 The Mechanical Derivation of Concurrency in Programs

Principal Investigator: Christian Lengauer

A Pascal-like program can easily be translated into a sequential execution, but often there is room for relaxations of sequencing. For example, while the two assignments

\[ y := x; \quad z := y \]

must be executed in sequence,

\[ x := x + a; \quad x := x + b \]

may be executed in either order, and

\[ x := a; \quad y := b \]

may also be executed in parallel. With a suitable formal semantic description of the program, the easy sequential execution can be "massaged" into a fast parallel execution.

This approach regards concurrency as a matter of program optimization, not program design. Such a view is suitable if we are only interested in the results of program executions (as in data processing or numerical applications) and not in the behavior of program executions (as in distributed or operating systems).

Viewing concurrency as program optimization has the benefit that we do not have to worry about it in the program design phase. We can develop and argue the correctness of programs without dealing with notions of concurrency such as processes, synchronization, and exclusion. After the program has been compiled into an easy sequential execution, we can concentrate on accelerating this execution (by putting in concurrency) without altering its semantics, which has been previously proved correct. The proof of correctness in the presence of concurrency is notoriously difficult. We only have to argue the preservation of correctness by the introduction of concurrency, which is much simpler.

In previous research [2, 3] I have defined a formal calculus for recognizing concurrency in Pascal-like programs. It splits the program development into

1. a design phase that produces a correct program which makes no explicit reference to sequencing, and

2. an optimization phase that takes an easy sequential execution (or so-called "trace") of the program and attempts to derive a semantically equivalent, sufficiently fast execution from it.

Recently I have begun to mechanize the optimization phase [4].

Massaging a sequential into a parallel execution means applying a recursive
transformation that is permitted by a set of formal transformation rules. These transformation rules can be derived from the semantic description of the program. The transformation of an execution could be performed by a suitable transformation algorithm. But since such an algorithm is only useful if it terminates, it can only transform a finite prefix of an unbounded execution. And a program whose length of execution depends on input, for instance, a program to sort an array of n elements (where n is the input), will be translated into an unbounded (recursive) execution.

To be able to deal with unbounded traces, we must view a trace transformation as an inductive theorem about the equivalence of two traces (the original trace and the transformation of it), not as a recursive algorithm that transforms the original trace element by element. An inductive proof can deal with an infinite trace in a finite time, whereas a recursive algorithm cannot.

For the coming two years I have the following plans:

1. Currently I am working on the implementation of the trace transformation calculus as it applies to a class of programs with particularly uniform semantics (sorting networks) in a mechanical logic and on several mechanical proofs of trace transformations. I need an appropriate collection of such proofs before I can proceed with further automation.

2. The next step will be the mechanical generation of intermediate information, i.e., auxiliary lemmas that are required for the mechanical proofs of the trace transformations studied in step 1, and others like them.

3. The third step will deal with the mechanical generation of trace transformation theorems, not just their proofs.

There are several directions one could take in the long term:

1. One could investigate program classes other than sorting networks, for instance, programs written in a general purpose programming language.

2. One could attempt the mechanical certification not of the trace transformations themselves but of the algorithms that identify useful trace transformations. The semantics of such algorithms will be much harder to formulate than the semantics of the trace transformations themselves.

This work contributes to a link between two research areas: programming methodology (the formal derivation of programs) and automatic theorem proving (the mechanization of logical calculi). Such a link is extremely important. Like the formal proof of programs, the formal derivation of programs and concurrency in programs will be feasible in a software production environment only if it is mechanically supported. Program logics in their present form are technically too intricate to be efficiently and reliably applied by hand on a large scale, and it is doubtful that they will become simpler in the future. The transformation of sequential program executions to
concurrent ones, as performed in the optimization phase of the outlined formalism, is an example of a derivation process that is conceptually simple (namely just an application of a few recursive patterns) but tedious (because induction is needed) and computationally complex (because the traces involved may be large) and, if executed manually, cumbersome and prone to error.

Today, theorem provers are already helping to certify the mass of relatively trivial verification conditions that are generated by current mechanical program verification systems. The proposed research is to study the mechanical treatment of more complicated verification conditions, conditions that are expressed recursively and proved by induction.

This research is particularly appropriate for our department for two reasons:

1. The central tool needed is an induction prover. Bob Boyer and J. Moore of this department are the creators and maintainers of the most powerful induction prover presently available [1]. Both J. and Bob have been very helpful in the initial phase of this project, and particularly their experience in the mechanical treatment of induction will be extremely valuable in the future.

2. We have exceptionally strong research groups in both mechanical theorem proving and the formal semantics of concurrency and already one project that links the two areas: the Gypsy language group of Don Good. Gypsy treats concurrency as a concern of program design. Incorporating my opposite view of concurrency as program optimization would complement our experience in the mechanical treatment of concurrency.

References


2.10 Integrating Knowledge into Problem Solving Search Procedures

Principal Investigator: Vipin Kumar

2.10.1 Introduction

Integration of domain-specific knowledge in problem solving procedures is very important. Obviously, some minimum amount of problem information or knowledge (which unambiguously specifies the problem) has to be present in a procedure before it can even try solving the problem. But experience tells us that this very minimum information is not enough to solve the problem in a reasonable amount of time. For example, resolution-based theorem provers have been unsuccessful in making useful inferences in practical domains unless they are augmented with domain-specific knowledge [5]. Similarly, mere specification of a problem in terms of state-space or And/Or graph search does not necessarily mean that the problem is practically solved; often the search spaces are very large, and problem-specific knowledge or heuristics are required to constrain search. Traditionally, knowledge representation has been studied independent of the problem solving search paradigm. But we strongly believe that the two issues are very much related.

In this research we are concerned with problems that can be stated as follows: Given (an implicit or explicit specification of) a set of objects X, find an element of X which satisfies a given property. A very large number of problems in planning, theorem proving, game playing, pattern recognition, decision making and puzzle solving can be formulated in this fashion. For example, in theorem proving we are given a set of axioms and a theorem to prove. The axiom set implicitly specifies the total set of proof trees which can possibly be generated from the axioms. The objective is to find from this set a proof tree for the given theorem.

An interesting and large subset of such problems can be considered as discrete optimization problems; i.e., they can be stated as follows:

Given a discrete set X and a cost function f defined over the elements of X,
find a least cost element of X.

In most of the problems of interest, X is too large to make the exhaustive enumeration for finding an optimal element practical. But a number of search procedures in different areas have been developed which use various heuristics and short cuts to efficiently search for an optimal element of X; for example, Alpha-Beta [12], SSS* [24], and B* [3] in minimax search; A* and other heuristic search procedures in state space search [1], [20]; AO* [20] in problem reduction search; and various branch-and-bound (B&B) [19], [11] and dynamic programming (DP) procedures [2], [6] in combinatorial optimization. Although many of these procedures have been thought to be related to each other, considerable confusion has existed. This is illustrated by a number of confusing and often contradictory statements in the literature regarding their true nature and interrelationships (see [14], pp.3-8)
2.10.2 Previous Work

In [14], [17] we have presented a general framework for representing problem specific knowledge in discrete optimization problems. In the context of this framework, most of the aforementioned procedures can be viewed as searching for an optimal element using problem specific structure present in X and f. This view reveals the true nature of these procedures, and clarifies their relationships to each other. It also aids in synthesizing (by way of analogy, suggestions, etc.) new variations as well as generalizations and parallel implementations of the existing search procedures [16], [18], [15], [17].

In our model, the discrete X is represented as a set of parse trees generated by a context-free grammar, and the cost function f is defined in terms of cost attributes associated with the productions of the grammar. Using certain relationships between context-free grammars, And/Or graphs and game trees, we have shown that a large number of search problems in Artificial Intelligence and in other areas of computer science can be formulated in this model.

Our model has greatly enhanced our understanding of the nature and interrelationships of dynamic programming, branch-and-bound and heuristic search procedures, and has provided us with a general paradigm for formulating and analyzing a number of search procedures. In the light of our model these procedures can be viewed as if they are searching for an optimal element of X in either top-down or bottom-up fashion by making use of the problem-specific structure present in X and f. The structure in X is present in its description as a set of parse trees of a context-free grammar. The structure in f is present in its definition in terms of the attributes associated with the productions of the context free grammar. Additional problem specific knowledge is used in the form of heuristic bounds associated with subproblems or subsets of X.

Top-down procedures can be considered as B&B procedures, and bottom up procedures as dynamic programming computations. A large number of heuristic search procedures (A*, AO*, alpha-beta, SSS*, B*) can be naturally grouped into one of the two classes. This classification explains most of the confusion regarding the nature and interrelationships of DP, B&B and heuristic search procedures.

The model has helped us devise general (top-down and bottom up) search procedures which subsume a large number of well known search procedures. For example, a top down (B&B) formulation for searching And/Or graphs subsumes a number of heuristic procedures such as AO*, SSS*, alpha-beta, and B*, and uncovers hitherto unknown interrelationships of these procedures (see [16], [14]).

2.10.3 Proposed Research

The general search procedures we have developed make use of two types of information to find efficiently an element of a discrete set X - "syntactic" and "semantic". The syntactic information is present in the representation of X and f (this information is used
in DP in the form of principle of optimality, in B&B in the form of a dominance relation, and in heuristic search procedures in the form of representations like And/Or graphs, state space, etc.). But the only semantic information used in these procedures is in the form of heuristics or bounds associated with subproblems or subsets of X. We propose to explore other types of problem specific knowledge (both semantic and syntactic) that can be integrated into these search procedures or their variations. By choosing models more general and richer than context-free grammars, it seems possible to integrate more problem specific knowledge into the model. This is expected to help reduce the complexity of search greatly.

It is instructive to note that many AI problems formulated in terms of our model (i.e., the composite decision process) are NP-hard [23]; hence no search algorithm is likely to solve them in their full generality in polynomial time. Thus introducing more domain-specific knowledge into the model (thereby restricting the generality of the problem) seems to be the only hope for their efficient solution. We speculate that this will also shed new light on the long standing question of whether P = NP, and suggest ways of dealing with NP-hard problems.

Our model is useful for those problems in which we need to find an optimal solution from a (implicitly or explicitly defined) set of solutions. As mentioned previously, there are a number of problem domains where we do not need to come up with a best solution; instead, any solution will suffice (e.g., in theorem proving, we need to come up with any proof of a theorem regardless of the size of the proof tree). We propose to investigate extensions and modifications of our model to such problems.

One straightforward way of extending our model to such search problems would be to define a binary \{0,1\} cost function on the elements of the discrete set X; i.e., for an element x in X, \(f(x)=0\) if x is acceptable as a solution else \(f(x)=1\). Then a procedure searching for a least cost element will find an acceptable solution if one exists. In fact, the backtrack technique [20, 10] which is quite often used for such problems can be considered in our framework as a top-down (B&B) depth-first search.

A major drawback of the backtrack technique is that the problem knowledge is used only in the description of X; the search strategy remains completely blind and exhaustive. However, in some cases, if enough problem-specific knowledge is present, then backtracking can be avoided altogether (e.g., [9]). Many attempts in different domains have been made to make backtracking less exhaustive (e.g., [22, 4, 25, 21, 13]). GPS [7] type means-ends analysis (as for example used in STRIPS [8]) is often quite helpful in constraining search but is limited in applicability. We will analyze the techniques used in all these attempts, and try to develop a unifying model which will facilitate systematic representation of knowledge and its use for efficient search.

A general paradigm for representing and using knowledge is expected to serve as a foundation for expert problem solving systems.
2.10.4 References


2.11 Parallel Algorithms and Architectures for Problem Solving Search

Principal Investigator: Vipin Kumar

The aim of this research is to develop parallel algorithms for problem solving tasks in Artificial Intelligence, and to develop special purpose VLSI architectures suitable for AI problem solving tasks.

2.11.1 Review of Previous Work

The unified approach to problem solving search procedures that we have developed in [18] has helped synthesize parallel implementations of a number of search procedures. In particular, we have developed two general schemes for performing B&B search in parallel. In the first approach, multiple processes perform depth-first search concurrently. At any time, at least one process has the property that if it terminates it returns an optimum solution; the other processes conduct a look-ahead search. In the second approach, the total search space is divided into several disjoint parts and each part is searched concurrently in a depth-first manner by a different process. Both approaches require very little inter-process communication and, therefore, are ideally suited for implementation on loosely coupled distributed systems (e.g. HXDP [14], cm* [29]). This is significant because tightly coupled multiprocessors like c.mmp [31] are hard to build for large numbers of processors; the complexity of the crossbar switch grows with the square of the number of processors involved. On the other hand, loosely coupled systems like cm* [29] can easily be built for thousands of processors.

We have used these schemes to implement parallel B&B formulations of SSS*, a versatile algorithm having applications in game tree search, structural pattern analysis and And/Or tree search. We have investigated the performance of parallel SSS* in the context of game tree search. A number of parallel game tree search procedures have been proposed and implemented [2], [10], [1], [23]. Simulation results [19] show that our parallel implementations of SSS* perform quite well in comparison to them.

2.11.2 Further Work on Design of Parallel Algorithms

Our parallel B&B formulations are applicable in principle to any problem that can be solved by the B&B technique. But the effectiveness of parallel implementations is expected to be dependent upon the nature of the specific problem being modeled. We propose to apply them to a number of domains. In particular, our parallel B&B formulation of the And/Or tree search procedure SSS* can easily be modified to be applicable to many other And/Or graph search problems. This is significant because And/Or graph models are extensively used in such wide domains as pattern recognition [15], [28], theorem proving, planning, and decision making. We plan to investigate the effectiveness of our parallel formulations in these problem domains.
A number of parallel algorithms for AI problems (e.g., [4], [27], [9], [16]) and other applications (e.g., [13], [24], [6], [8], [25], [12], [17]) have been developed. We propose to investigate the effectiveness of these parallel algorithms and their variations to the problems of our interests.

2.11.3 Interprocess Communication vs. Speedup

Our parallel B&B formulations for And/Or tree search require very little interprocess communication because the semantic knowledge used by the B&B procedure can be expressed simply as a real value (a heuristic bound) to be communicated among the cooperating sequential processes. If we develop B&B procedures that use richer representations of knowledge than in the parallel version, more information sharing is possible among the concurrent processes. Individual (parallel) processes can now perform search more efficiently because more information is available to them. But the advantage derived from increased availability of information may be nullified by the increased communication overhead, particularly in loosely coupled systems. Thus, how much interprocess communication should be permitted in a parallel algorithm is an important question [22].

We propose to study this tradeoff between the amount of communication permitted and the speed-up achieved for a spectrum of architectures ranging from loosely-coupled to tightly-coupled systems. This study will be facilitated by the parallel processor TRAC, being built at the University of Texas [26]. TRAC, the Texas Reconfigurable Array Computer, consists of 16 processing units connected to 81 memory modules via a dynamically reconfigurable banyan switch network. By changing the degree of the banyan switch, TRAC-type systems can be built to provide desired degree of interprocessor communication. Furthermore, TRAC can dynamically reconfigure its processor, memory and I/O resources to implement a variety of parallel computation architectures, which makes it a very convenient environment for testing the performance of parallel algorithms.

2.11.4 Special-purpose Architectures for Problem Solving

With the advent of VLSI technology, it has become feasible to construct large parallel processing systems as long as they are made of fairly regular patterns of simple processing elements. This makes it possible to develop special purpose architectures for performing problem solving tasks. Japan's Fifth Generation project has accelerated research on computers that embody knowledge bases and support problem solving and inference functions. Due to the tremendous computing power required by AI applications, these systems are expected to use extensive parallelism.

A number of systolic architectures (e.g., [20], [32], [11]) and tree searching machines (e.g., [7], [3], [21]) have been proposed, and some of them have been designed and built. A number of general purpose, inherently parallel architectures are being investigated as
alternatives to the traditional von Neuman architectures. These new architectures can be grouped into two categories: data driven (e.g., data flow systems [5]); and demand driven (e.g., reduction machines [30]). In the context of this categorization, von Neuman architectures can be considered as control-driven [30]. While developing a unified approach to problem solving search procedures [18], we have shown that a large number of problem solving search procedures can be grouped into two categories: top-down search and bottom-up search. It is interesting to note that data-driven architectures are naturally suited for bottom-up procedures, and demand-driven architectures are naturally suited for top-down procedures. We propose to investigate this relationship further, and evaluate the suitability of the data-driven, demand-driven and various special purpose architectures to problem-solving tasks.

2.11.5 References


12. T. Herman and K. M. Chandy, A Class of Distributed Search Algorithms, Tech. report, Dept. of Computer Sciences, Univ. of Texas at Austin, Austin, TX78712, 1983.


19. V. Kumar and L. N. Kanal, Parallel Branch and Bound Formulations for And/Or Tree Search, submitted for publication. 1983.


2.12 Discourse Representation and Processing

Principal Investigator: Carlota Smith

We propose a study that will advance the application of syntax and semantics in Artificial Intelligence systems. Our theory, Discourse Representation Theory, presents a novel, integrative solution to the problem of parsing and interpreting sentences of natural language. The theory (Kamp 1981) interprets sentences both as individual units of communication and as parts of an extended discourse. This type of interpretation will be an essential part of any computational theory that deals, directly or indirectly, with natural language.

Discourse Representation Theory (DR theory) combines sophisticated semantic and syntactic analysis with a computational approach: it is therefore of particular interest. The interpretation of a series of sentences is built up formally from two sources: a syntactic parsing algorithm that maps the sentences into syntactic structure, and a representation construction algorithm that maps syntactic structures into semantic representations. These representations are assigned truth conditions.

The problem of the analysis of discourse has long been a stumbling block for work in artificial intelligence and linguistics. Attempts to use sentence-based approaches to discourse have not been successful. DR theory can deal with pragmatic information and inferences, as well as with the syntactic and semantic structures that underlie the separate sentences of a discourse. It is thus equipped to handle areas on which previous attempts to handle discourse have floundered.

This work draws on several disciplines. We will develop the theory and its computational implementation, investigating the problems that arise. Two computational implementations of the Discourse Representation algorithm already exist. Lauri Karttunen, a member of our project, has developed a computational implementation of the theory using LISP; Rohrer has developed an alternative system using PROLOG. We intend to work with these implementations, extending their application to our new material. We also intend to combine the construction algorithm with a parser employing Chomsky's Government and Binding syntactic theory (Chomsky 1981). The combination of these techniques will enable us to construct a powerful system capable of analyzing and, eventually, generating the full range of natural language structures. Our research will thus make use of frequent experimentation involving computer implementation.

The theory has already been worked out in detail for anaphora as well as tense and aspect, both extremely difficult areas (Kamp 1981, Smith 1980, 1983). We are currently developing general representation algorithms for plurals, generalized quantifiers and clausal complementation. We propose to extend the algorithm to incorporate more complex syntactic and semantic phenomena. Further, we plan to integrate into our theory a general theory of inference capable of dealing with presupposition, implicature
and inferences based on world knowledge as well as entailment. Solution to these problems is fundamental to further progress in developing question answering, expert and machine translation systems that must understand or generate natural language.

The University of Texas at Austin has major resources in this and related areas. The Linguistics Department is outstanding: it is ranked in the top three departments of the country, and it is particularly strong in the formal areas that are essential to this research. We also plan to draw on interested people from the Departments of Philosophy and Computer Science, and on the interdisciplinary environment of the Center for Cognitive Science. Primary personnel for this project will include Hans Kamp, the founder of Discourse Representation Theory; Carlota Smith, Chair of the Department of Linguistics, who has published widely in the syntax and semantics of natural language, and Nicholas Asher and Daniel Bonevac, who have also worked in DR Theory. Besides this nucleus we will consult frequently with faculty members from other departments, many of whom are working on related topics. These will include Lauri Karttunen, Robert Wall, Robert Boyer, and Robert Causey.

Inference is an essential element in natural language processing. It has been quite widely studied in recent years but is usually considered separately from other types of interpretation. This is a real handicap in natural language processing. We plan to integrate into DR Theory a general theory of inference capable of handling presupposition, implicature, and inferences based on world knowledge, as well as entailment. DR Theory offers a way to combine and model these complex processes formally. A computational implementation of a system of inference, together with a discourse representation algorithm, will give us a system with many artificial intelligence applications. In particular, it will be useful for question-answering systems, machine translation, user-friendly programs, and expert systems.

Central to DR Theory is a level of semantic representation that is distinct from the structure of individual sentences. DR theory makes no radical separation of the semantics of single sentences from the semantics of discourse, claiming that proper evaluation of a sentence usually requires analysis of the antecedent context. It views sentences as, semantically, mappings from contexts to contexts (Kamp 1981a, Stalnaker 1975).

We shall focus our research in two main areas, semantics and syntax.

2.12.1 Semantics

1. The Construction Algorithm and Model Theory. In extending the construction algorithm, we shall draw on preliminary results that have already been obtained by members of our research team. The semantic treatment of complex clauses is being investigated by Bonevac, with special emphasis on clausal complementation; Asher 1983 and Bonevac 1983 deal with belief sentences and other verbs of mental state, and Asher
has already developed a semantics for belief meeting the basic semantic criteria and appropriate for integration into discourse representation structure.

2. Speech Acts. Many problems in this area share an intersentential character; DR Theory is thus uniquely suited to their solution. By constructing the meaning of a sentence as a function from antecedent to resultant contexts, DR Theory incorporates and formalizes concepts that have so far resisted precise analysis. Current work indicates that DR theory is able not only to distinguish two types of presupposition, but will also generate pragmatic presuppositions. DR theory, then, contains a far richer set of concepts pertaining to speech acts than other semantic theories.

3. Logic. Standard logical systems, developed primarily for use in mathematics, handle only a limited portion of natural language. DR theory obtains a far wider range. Because it uses first-order models, DR theory can readily apply the sophisticated techniques of modern model theory. Images of DRSs under embedding functions are, in effect, situations in the sense of Barwise and Perry (1983); they correspond to first-order models (Kamp 1983). Kamp has developed rules of inference for a fragment of DR theory that uses only discourse markers and sentential operators to capture the full strength of first-order logic. Kamp’s work parallels resolution techniques developed by Boyer and others (Boyer and Moore 1979).

Any model of language comprehension requires not only a deductive logic but also a theory of plausible inference. We intend to deal with the latter by developing the notion of probabilistic reasoning. We require a system of probabilistic reasoning for default inferences, among other types. We shall adopt subjective partial conditional probability distributions, defined over subsets of DR conditions or DR structures (Asher 1982). This system will be compared with other forms of nonmonotonic reasoning such as John McCarthy’s method of circumscription (McCarthy 1980). We shall also ask whether these systems lend themselves to efficient computational techniques.

2.12.2 Syntax

The DRS algorithm converts syntactically analyzed structures into DRSs. This design assumes a prior step: the conversion of linguistic structures into the syntactic units on which the algorithm operates. We intend to explore the relation between syntax and DR theory and to develop the current syntactic parsing algorithm. The project has important ramifications for syntactic theory in general, since the relation between syntax and semantics is not well understood.

1. The form of the input to the DRS construction algorithm. We require a syntax that can feed directly into this algorithm. We will investigate the question of whether configurational information is necessary or desirable for determining the possibilities of anaphoric links. Government Binding theory, the syntactic theory that has most fully developed an approach to anaphora, determines anaphora at S-structure and depends on
configurational properties. A comprehensive theory of anaphora will probably incorporate such syntactic information. We will also investigate similar questions for quantification.

2. Syntactic rules: The relation between syntax and semantics. We will develop the syntactic rules necessary for the construction algorithm of DR theory. These rules will require that we take a stand on a classical question, that of the relation between syntax and semantics. DR theory provides a principled way of understanding this relation. We will of necessity delineate it in the construction of the rules for the theory; and we expect this detailed implementation to suggest an interesting solution to the classical problem.

3. Other contributions of syntax to discourse. We also intend to explore aspects of syntactic structure that are less well-understood, in their relation to discourse, than anaphora and quantifiers. In particular, we are interested in whether and how the structure of a sentence gives information about the importance of material, the focus of the discourse, etc. Further, we will look at the way the thematic relations of individual sentences play a role in the structure of the discourse.

2.12.3 References


2.13 Artificial Intelligence and Robotic Manipulators

Principal Investigator: N. Duke Perreira

2.13.1 Introduction

The goal of this project is the development of a working, intelligent, and adaptive hierarchically controlled robotic system. The system will be used to test and develop various artificial intelligence motion-planning algorithms in real world systems. Although the concept of hierarchical control of robotic systems is not new [Albus], robotic systems that incorporate only the most rudimentary forms of hierarchical control have as yet been developed. A highly sophisticated working system is essential to developing and testing motion-planning algorithms that are capable of intelligently determining the desired robot motion based on general verbal commands while being able to adapt to changes in the environment.

For the purposes of demonstration, robotic manipulators will be used in this research project. This type of system is believed to be generically similar to many of the types of devices in which the Army may have direct interest.

Shown in Figure 1 is a block-diagram representation of the levels in the hierarchical structure for a robotic system. The highest level communicates either directly or through a second supervisory hierarchy to a person or persons who desire the completion of the major task or tasks. The system is designed to alleviate the need for other forms of user interaction. The major task is decomposed into more detailed tasks using artificial intelligence techniques. These techniques take into account information derived from processed sensory data or reports from lower-level controllers and predictions generated by knowledge bases or world models. Each level of the hierarchy is in general composed of three sub-elements: task decomposition, a world model, and feedback processing, as shown in Figure 2. Some of the current and anticipated research projects in each of these areas are discussed in the introductory section on Robotics. These classes of sub-elements in turn define three sub-hierarchies. Each sub-element processes a stream of data while being continually adapted based on processed data from another sub-element. For example, the task-decomposition sub-element processes task description data into finer task details. Simultaneously, output from the feedback processing unit (essentially, processed sensory information of the environment) is used to modify or adapt the task-decomposition algorithm because of perceived changes in the environment.

This technique has been used to develop first and second level adaptive controllers to compensate for link flexibility [Perreira, Hudgins, 1982], and third and fourth level controllers for the case of obstacle avoidance [Perreira, Chapman, 1982]. The author is currently working on a project where second level controllers are used to decompose world trajectory data into joint space trajectory data while adaptively compensating for errors in the robotic system model, actuation measurements, and control action. [Work being conducted by Perreira, Colson, and Ibarra.]
HIERARCHICAL CONTROL LEVELS
FOR ROBOTIC DEVICES

7th Level

6th Level
Human Commander

5th Level
Major Task

4th Level
Simple Task

3rd Level
Elemental Move

2nd Level
World Space Trajectory

1st Level
Joint Space Trajectory

0th Level
Actuation

Environment

FIGURE 1
SUB-ELEMENTS OF A HIERARCHICAL CONTROL LEVEL ELEMENT

Higher Level or Commander

EVALUATION → DIRECTIVE

TASK DECOMPOSITION

Sensed Directive

FEEDBACK PROCESSING → WORLD MODEL

Deduced Directive

Decomposed Task

Lower Level

Sensory Feedback → Action

ENVIRONMENT

Figure 2
Each level of the hierarchical controller, in fact, each sub-element of any particular level, could be considered an independent processing unit. Ideally, these units would be interconnected through a memory unit common to both a higher and a lower level controller. Although this type of hardware is not currently commercially available, VLSI technology is sure to change this.

2.13.2 System Design and Components

The lowest levels of the task decomposition hierarchy are most readily implemented using digital adaptive servo-controllers programmed directly in assembly language or in languages, such as C or FORTH, that are easily implemented on different systems. The primary reason for this choice is the continual need to update the actuator inputs so that the robot motion is smooth; thus, minimal machine code and processing time are of major importance. In addition, transportability increases the benefit of the work effort spent on the higher level coding. The highest-level task decomposition controllers need to be programmed in a more sophisticated manner using languages such as LISP. In these controllers, although processing speed is still important, the critical consideration is intelligent adaptation using inferences. The middle-level controllers need to be programmed in languages that perform "number crunching" functions efficiently and with some level of intelligence. For robotic manipulators such as the IBM 7565 (RS/2), languages such as AML fill this need. AML is a Pascal-like language that contains many string-manipulation functions. Laboratory work during the last few months indicates that this manipulator can be directed to a pose that is adaptively changed during its motion by use of monitor commands from an external signal. The pose, or desired robot "tool" position and orientation, can thus easily be directed by a higher-level controller that does not also have to consider low-level control problems. The procedure for determining the directed poses and tasks will be one of the research goals of this project.

In some cases, sensory information should be processed using a processing system directly associated with a particular level of the hierarchical controller, while in some other cases an independent or set of independent processing units is required. For example, position and velocity sensing are rather simple operations with minimal computational burden. Thus, the actuation controller processing unit can easily accommodate the processes of determining a smooth joint actuation profile, while attempting to achieve this profile and simultaneously measuring the actual joint motion profile. On the other hand, when motion specifications require force, torque or impedance to be controlled either at the "tool" or at the joint, a separate processing unit may be required.

In the case of image processing, an array of high-speed processors may be required. For example, on the Automatix AV II image processing system, two Motorola 68000’s operating at 10MHz are required to achieve somewhat close to real time processing. The first processor is used to map the pixel intensities into a two-state black-and-white image using an intensity map input by the programmer. The first processor also determines
the blob-hole tree structure in a user-defined window using a Stanford Research Institute (SRI) algorithm. The second processor is used to determine the features of each of the blobs and holes. The image features of the AV II (located in the robotics research laboratory) include the location and orientation of the blobs, as well as their area, perimeter, various moments, etc. Intensity histograms and statistical correlations with a user-trained knowledge base can also be determined. To a lesser extent, RAIL, the AV II language, also allows the user to change program parameters adaptively through external commands. These commands may be generated by an intelligent algorithm.

In order to use the externally sensed data (for example, image data from the AV II) intelligently to control a robotic device (for example, the IBM RS/2), an independent processor capable of task decomposition at a higher level is required. In this project, a LISP machine will be used. In addition, world model data will be obtained from a VAX 750 using robot simulation programs.

Use of Artificial Intelligence

The proposed system will be used to examine and compare the use of various AI algorithms to control robotic devices. Use of the following techniques will be examined in the project.

1. Use of natural language interpreters to translate spoken commands (or keyboard simulations of spoken commands) into machine-intelligible major task commands.

2. Use of predicate logic as the basis of knowledge representation to put feedback processing data into a form that allows resolution to deduce the desired task decomposition.

3. Use of various forms of structured knowledge representations.

4. Use of game theory in the higher levels of the hierarchy to adapt the task decomposition to changes in the environment.

5. Use of procedural logic to determine the order of task performance.

6. Use of learning by parameter adjustment in various levels of the hierarchy, especially at the lowest levels, to modify the world model.

2.13.3 References


8. Ibarra, R., and Perreira, N.D., Techniques to Improve the Accuracy of Robotic Manipulators. This work is being performed as part of the first author's Master's Thesis. To be submitted to Amer. Soc. of Mech. Engr. 1984.

9. Smith, P. and Perreira, N.D., Real Time Pose Recalibration of Robot Manipulators for Improved Accuracy. This work is being performed as part of the first author's Master's Thesis. To be submitted to the Amer. Soc. of Mech. Engr. 1984.

10. Mehta, A. and Perreira, N.D., Adaptive Task Decomposition for Assembly and Disassembly Operations Performed by Robotic Manipulators. This work is being performed by the first author as part of his doctorate work. It is being sponsored by the U.T. Center for Fusion Engineering. The work will be submitted to the Inst. of Elect. and Elect. Engr. for publication.

11. Nannayakarra, S. and Perreira, N.D., Recursive Hamiltonian Canonical Equation Formulation for Robotic Manipulator Control. Work performed as part of a graduate level robotics course.

### 2.13.4 Equipment Needs

Because the robotics research proposed here requires special equipment, we have included this section describing and justifying the proposed equipment acquisitions.

An Automatix AV IV image processing system will be needed to supplement the current AV II vision system, which is already fully utilized. The AV IV contains more powerful hardware and software than the AV II system.
<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV IV</td>
<td>$32,500</td>
</tr>
<tr>
<td>Memory Module</td>
<td>5,500</td>
</tr>
<tr>
<td>4 Cameras</td>
<td>13,600</td>
</tr>
<tr>
<td>4 lens extensions</td>
<td>400</td>
</tr>
<tr>
<td>8 lenses (50, 25, 12 mm)</td>
<td>1,200</td>
</tr>
<tr>
<td>Interface Module</td>
<td>1,000</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>$54,200</strong></td>
</tr>
</tbody>
</table>

Assorted hardware, such as digitally controlled camera tripods, robot grippers, and other sensors will be used to enhance our external sensory capabilities.

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 digitally controlled tripods</td>
<td>$5,000</td>
</tr>
<tr>
<td>robot gripper for Puma</td>
<td>14,000</td>
</tr>
<tr>
<td>assorted sensors and hardware</td>
<td>10,000</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>$29,000</strong></td>
</tr>
</tbody>
</table>

A LISP machine dedicated for use with the robots and image processing machines will be used to process the artificial intelligence algorithms. This machine will require special interface modules resulting in a higher than average cost.

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISP Machine</td>
<td>$110,000</td>
</tr>
<tr>
<td>Special Interfaces</td>
<td>20,000</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>$130,000</strong></td>
</tr>
</tbody>
</table>

An additional one or two robots may be needed depending on the number of students involved in the program. The first of the two to be purchased would be a "SCARA" arrangement. This arrangement is well suited for the wear-and-tear expected by machines in classroom use. The second machine would be a Puma 260, a small version of the 560. These two robots should handle from five to ten classroom users and the additional two to five researchers who are expected to be part of this program.

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCARA type robot with controller</td>
<td>$40,000</td>
</tr>
</tbody>
</table>
Puma 260 robot 30,000.

TOTAL $70,000.

The costs identified above and summarized below reflect a program with approximately ten students doing intensive course work and some amount of research in artificial intelligence and robotics. The equipment, in addition to current laboratory equipment, could handle up to 30 additional students taking one or two robotics courses as part of their artificial intelligence program but not intending to do any research in the area.

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vision Hardware</td>
<td>$54,200.</td>
</tr>
<tr>
<td>Assorted Hardware</td>
<td>29,000.</td>
</tr>
<tr>
<td>LISP Machine</td>
<td>130,000.</td>
</tr>
<tr>
<td>Robotic Manipulators</td>
<td>70,000.</td>
</tr>
</tbody>
</table>

TOTAL $283,200.
2.14 AI in Target Recognition and Tracking

Principal Investigator: J. K. Aggarwal

An important task for the Department of Defense is the processing, analysis and understanding of sequences of images for the recognition and tracking of objects. After the initial digitization of the images and subsequent removal of noise, the segmentation of the individual images must be performed. Normally, segmentation yields the targets to be recognized and tracked. It may happen that the segmentation process itself requires pairs of images to isolate the objects of interest in the image. Our group at the University of Texas has conducted research in various aspects of recognition and tracking of targets as described in [1-6]. At the recent NATO Advanced Study Institute held in Braunlage, Germany and specifically devoted to Dynamic Scene Analysis and Image Sequence Processing, our group reviewed and presented the relevant results [7]. The techniques developed so far are based upon mostly statistical rules.

These rules work fine for laboratory-generated synthetic images. In a more realistic domain where the images are extremely noisy, the decision process may have to depend on lengthy sequences of images. Also, one may have to maintain several alternate interpretations or descriptions of the observed objects in the scene. A "correction" phase of the system may be invoked periodically to eliminate less likely alternatives and resolve conflicts between interpretations. Or it may happen that the observed images in the sequences may themselves resolve the conflicting interpretation; or the more serious situations may arise were the system made an erroneous decision and must backtrack to correct the erroneous decision; or the system may not be able to make a decision until more evidence becomes available.

The objective of the proposed investigation is to develop a knowledge-based recognizer and tracker for a mobile robot system with several specific overall aims:

1. Intelligent tracking of targets with occlusions and evasive maneuvering.

2. Organization of knowledge bases for real-time processing of image sequences.

3. Optimization of image processing with respect to paging and use of memory hierarchies, including multi-leveled caches and content-addressable memories.

4. Mathematical techniques for minimization of logical inconsistencies developing in semantic image interpretation schemes based on the integration of local structures. Of particular interest will be optimal control methods for integration of image sequences.

5. Complexity of interconnectivity between major knowledge groupings in partitioned knowledge bases.
2.14.1 References


2.15 Automated Translation and Natural Language Processing

Principal Investigator: Winfred P. Lehmann

2.15.1 Introduction

Long-term success in Natural Language Processing depends on finding tools that work for the broad range of language types. Investigators must undertake a similarly broad range of language study, covering at least one representative of each major language type as defined by Greenberg [78]: SVO [subject-verb-object] languages (e.g., Chinese, English), representing approximately 40% of human languages; SOV languages (e.g., Japanese), representing approximately 50% of human languages; and VSO languages (e.g., Arabic), representing approximately 10% of human languages. These groups of languages are fundamentally different in many if not most aspects of their behavior. The structural differences alone are of critical importance [Lehmann, 78]. But text phenomena play a large role in Natural Language Processing applications, and these have not been subjected to the necessary scrutiny; de Beaugrande and Dressler [81], and also Kittredge and Lehrberger [82], represent a start (for SVO languages only), and in addition demonstrate the fundamental necessity for increased study of this area. Then, too, handling grammatical errors is important in any practical application, yet few workers have explored this area and little knowledge has accumulated.

Automated Translation (AT) of natural human languages is not a subject about which most scholars feel neutral. This field has had a long, colorful career, and boasts no shortage of vociferous detractors and proponents alike. During its first decade in the 1950’s, interest and support were fueled by visions of high-speed high-quality translation of arbitrary texts (especially those of interest to the military and intelligence communities, who funded AT projects quite heavily). During its second decade in the 1960’s, disillusionment crept in as the number and difficulty of the linguistic problems became increasingly obvious, and as it was realized that the translation problem was not nearly so amenable to automated solution as had been thought. The climax came with the delivery of the National Academy of Sciences ALPAC report in 1966, condemning the field and, indirectly, its workers alike. The ALPAC report was criticized as narrow, biased, and short-sighted, but its recommendations were adopted (with the important exception of increased expenditures for long-term research in computational linguistics), and as a result AT projects were cancelled in the U.S. and elsewhere around the world. By 1973, the early part of the third decade of AT, only three government-funded projects were left in the U.S., and by late 1975 there were none. Paradoxically, AT systems were still being used by various government agencies here and abroad, because there was simply no alternative means of gathering information from foreign [Russian] sources so quickly; in addition, private companies were developing and selling AT systems based on the mid-60’s technology so roundly castigated by ALPAC. Nevertheless, the general disrepute of AT resulted in a remarkably quiet third decade.
We are now into the fourth decade of AT, and there is a resurgence of interest throughout the world -- plus a growing number of AT systems in use by government, business and industry. Commercial firms are also beginning to fund AT R&D projects of their own; thus, it can no longer be said that only government funding keeps the field alive (indeed, in the U.S. there is no government funding, though the Japanese and European governments are heavily subsidizing AT R&D). In part this interest is due to more realistic expectations of what is possible in AT, and realization that AT can be very useful though imperfect; but it is also true that the capabilities of the newer AT systems lie far beyond what was possible just one decade ago.

The Linguistics Research Center, developer of the highly successful METAL system for Automated Translation, has been a world leader in the field in recent years. The most important ingredient missing from our work in Automated Translation has been long-range research in the areas mentioned in the first paragraph above. We propose to rectify this situation, while not entirely abandoning our orientation toward practical application. In particular, we propose to develop, implement, and test advanced theories of Natural Language Processing (NLP), within the overall framework of the METAL Automated Translation (AT) system. These theories will concern not only the structural characteristics of the several languages to be considered, but also their textual characteristics and the pragmatics involved in their use.

2.15.2 Past Linguistic Research Methodology

Linguistic study in the past has concentrated on the analysis of individual sentences. Moreover, one of the leading schools of linguistics limits itself to the study of "an ideal speaker-listener, in a completely homogeneous speech-community, who knows its language perfectly and is unaffected by ... grammatically irrelevant conditions ... in applying his knowledge of the language in actual performance" [Chomsky65, p. 3]. By contrast, processing natural language by computers requires that one deal with actual, not ideal languages. It must concern itself with language in text form, not as isolated sentences. The texts will include imperfections, involving grammatical, spelling, and other errors. In short, NLP requires attention to language as it is used, whether in technical texts, conversations, or more formal writing.

2.15.3 Typical NLP Research Methodology

Most current work in NLP falls under the headings of "semantic models," "discourse structures," and/or "text coherence." The theories underlying these areas relate to such applications as: resolution of anaphora (determining what nouns and pronouns refer to); determining prepositional phrase attachment (i.e., which noun or verb or other linguistic structure a prepositional phrase most directly modifies); scoping of quantifiers and conjunctions (identifying another kind of attachment); resolution of ambiguity (determining the intended meaning of a word or phrase); and discovering coherence relations (e.g., how sentences go together to form a meaningful text, or conversation).
From time to time, significant new techniques are proposed to solve one or more of these problems. It is usually the case that a litany of supporting cases and counter-examples pervade the literature (both kinds composed of sentences invented on-the-spot by the writers), with little or no indication of the statistical likelihood of success in actual practice. Theories might be discarded for failing to account for a single class of examples, with little or no thought given to the relative incidence of such failures. This is in part due to the concentration of most computational linguists on armchair experimental methods (a paradox of sorts, with computers so readily available to them), and a desire for *linguistic purity* in their theories. But it is perhaps mostly due to the related fact that few groups anywhere have developed a grammar of sufficient coverage that raw texts can be fed into a system and meaningful measurements taken of its qualitative performance.

2.15.4 The R&D Potential of METAL

One of the particular strengths of the METAL system has been its potential for accommodating a variety of linguistic theories/strategies. For example, our German analysis module is based on a context-free phrase-structure grammar, augmented by procedures including arbitrarily-powerful transformations [Bennett82]; our English analysis component, on the other hand, employs a modified GPSG approach [Gazdar81] and makes no use of transformations. The differences between these two approaches represent nothing more than the different personal preferences of the linguists in charge of the respective grammars. These approaches exist side-by-side within the identical software environment [Slocum81a]; other linguistic approaches are allowed as well (e.g., Lexical-Functional Grammar [Bresnan77]).

A twin strength of the METAL system is its accommodation of a variety of computational strategies. In recent years, for example, no fewer than a dozen variations on three major parsing strategies for the METAL system have been submitted to detailed empirical analysis in large-scale experiments (e.g., [Slocum81b]). The implementations of these parsing programs and their strategy variations did not require the modification of the existing grammars (though taking the best possible advantage of two of the strategy variations did entail slight changes in some rules). As another example, two different methods for implementing the transformation facility have been used, without changing any grammars -- and in fact, the two methods co-exist, the operative one depending on details of transformation invocation invisible to the linguist.

Finally, the METAL system is highly modular, having been carefully designed to support a large variety of linguistic models -- and evolution in those models -- in anticipation of a continually maturing theory of language. A considerable amount of evolution has indeed taken place; thus, METAL has already demonstrated a capacity for hosting different computational and linguistic theories in a growing R&D environment. METAL is also a highly portable system, having been implemented in UCILISP to run on a DEC-10, in INTERLISP to run on a DEC-20 and a VAX, and in ZETALISP to run on the Symbolics LM-2 and 3600 Lisp Machines.
2.15.5 Natural Language Processing using METAL

METAL is a complete natural language processing system, rather than a basic "translation engine." A few parts of the METAL system are specific to the translation application; for example, there is a software package for text processing that formats the output translations like the original input documents. Most components, however, are packages of general utility for Natural Language Processing. In addition to the general parser, and a new generator under development, we have, for example, developed: a database management system (for dictionary entries and grammar rules); a rule validation module (to eliminate most human errors in dictionary entries and grammar rules); general-purpose dictionary construction software (to enhance human efficiency in coding lexical entries) [Slocum82; Slocum and Bennett, 82]. METAL software exists to eliminate a lot of drudgery for its developers as well as its users.

Other applications of METAL -- for example, a natural language interface to database systems -- could be devised using the same software, grammars, and dictionaries. The METAL system is a highly efficient natural language processor, with very large dictionaries (currently 10,000 words) and grammars (hundreds of rules) in multiple languages (currently English and German, with more to come). The Linguistics Research Center is not seriously limited by grammatical/lexical coverage problems with respect to German or English, and expects to develop similar coverage for other languages in the near future -- including the languages proposed in this project. As a result, the LRC stands almost alone in its ability to submit modern linguistic theories to empirical validation, and Natural Language Processing (NLP) applications to use in a practical setting, in several languages. Only a handful of organizations in the world -- and no others anywhere in the U.S. -- can make such a claim.

By testing linguistic theories in a wide variety of settings, we are able to determine the differential utility of techniques depending on the language and type of text being considered. We already know that some problems in some domains may not require general solution in other domains (e.g., noun anaphora in translation). We have reason to suspect that many NLP techniques are specific to English (since almost all of them are developed in the U.S., in the context of English alone); that some are more general (working reasonably well for, e.g., the Indo-European family of languages); and that very few are widely applicable (to, e.g., such dissimilar languages as English, Chinese, Arabic, and Japanese). We also expect to increase our knowledge about text "sublanguage" [Kittredge and Lehrberger, 82] by deriving empirical characterizations of the differential linguistic properties of texts in specific domains.

2.15.6 Proposed Research using METAL

The Linguistics Research Center occupies a unique position in the U.S. for its ability to devise and/or test general theories of Natural Language Processing. We therefore propose to embark on a 5-year program of research relating to the analysis, translation,
and synthesis of multiple languages, by testing state-of-the-art theories developed by linguists here and elsewhere in order to measure and report the empirical properties of the languages and all the computational techniques tested on them. As primary natural languages for testing these theories, we propose English, Arabic, Chinese, and Japanese. As secondary host languages, we can draw upon our existing and developing systems for German and Spanish.

Our proposal, therefore, is to greatly expand our long-range research program (as opposed to our current commercial development effort) by mounting a serious effort to formulate and implement new computational models of actual linguistic phenomena at all levels (from word to text), and then test these in large-scale settings in multiple languages. The range of phenomena (i.e., linguistic theories) to be considered will concentrate on those related to technical texts, though other types of material will be studied as well.

Ultimately our focus on translation is not a limiting factor in research since every linguistic problem will eventually crop up; indeed, our concentration on the translation of large volumes of text has brought to our attention more linguistic phenomena than armchair methods could ever devise. But in practice, there are problems which occur relatively frequently in translation, and there are others which occur relatively infrequently. We will allow the natural incidence of (technical) translation problems to guide our search for categories of computational solutions. We already know, for example, that the resolution of pronominal anaphora lies near the top of the translation problem priority list, while the resolution of other kinds of anaphora is much less important (for translation purposes). Other examples of task priorities could be given, but it is enough to note that the incidence of problems in natural texts, as opposed to made-up examples, will be our guide. By concentrating on natural texts in multiple, radically different languages, we will gain a clearer picture of the potential for practical applications of computational linguistics theories than could ever be gained by mere academic conjecture.

2.15.7 References


2.16 Statistical Metaphors and the Computer Medium

Principal Investigator: Joseph G. Deken

2.16.1 Introduction

"Statistics" is defined here as "quantitative communication about data." Statistics is viewed as an essential component of the scientific process, defined as the "development of communicable, empirically accurate models." In the proposed research, we will investigate statistical metaphors and computing environments which enhance human understanding of and communication about data by appealing to natural human sensory and linguistic metaphors. Computer systems that support these metaphors will not be "expert system" consultants about traditional statistical methods, but rather ensembles of intelligent subsystems that enable data to describe themselves automatically and interactively to a human investigator.

The development of these metaphors implies a concurrent investigation of the computer as an active medium, utilizing its potential to make the new metaphors effective. The target will be metaphors that function with current computing technology and hold near term promise for efficient implementation with more advanced systems.

2.16.2 Dimensions of the Proposed Research

The role of the computer as medium is threefold. It is a:

i) Communications medium: Computer networks allow raw and processed data, models and the programs that describe them to be transported and implemented nationwide. Via network distribution, this project will promote discussion among statisticians nationwide of the desirability and feasibility of new metaphors, as well as provide examples of new ways of analyzing interesting data sources.

ii) Modeling medium: The computer is much more versatile as a modeling medium than the human memory for digits or traditional paper and pen. Consequently, the models used to represent data can be far more ambitious than summary numbers or straight-line fits. In particular, we will explore ACTOR-based metaphors. Through these metaphors, we will go beyond the "expert-system" view of developing a computerized statistical consultant. The goal is to create an intelligent statistical environment, in which ACTOR-based data and models literally explain themselves.

iii) A nutrient medium: The computer functions in a research environment as a life-support system for cultivating ideas. Groups at Stanford (projection pursuit, Lisp machine statistics environment) and Bell Laboratories (s-language project) have benefited from this phenomenon. An effective artificial intelligence environment will undoubtedly prove equally productive at the University of Texas, especially as it focuses the attention of the University's Center for Statistical Sciences, a campus-wide research group of faculty experts in all aspects of theoretical and applied statistics.
2.16.3 An Illustrative Metaphor: Actor-Based Fitting

The theory and practice of statistics, since R. A. Fisher, is strongly oriented towards geometric metaphors for describing data. In assessing the geometric orientation one finds a significant advance over previous approaches. (Computationally, the projection of a Euclidean data vector onto orthogonal linear subspaces contains the same information as the strictly algebraic manipulations of the older, sums-of-squares decomposition. By contrast though, the geometric metaphor suggests further insights and alternative analyses.) The success of the geometric approach suggests as well that if other new metaphors of equal intuitive content could be found and managed, they would be similarly productive.

For example, a standard statistical fitting procedure would interpret a set of data in terms of an overall (geometric) model, and departures (residual vector) from that model. By contrast, an ACTOR based metaphor would consider DataObjects, Organizers, and Reporters. DataObjects collect and update information to influence the Organizers. Organizers attempt to configure the DataObjects for minimum *stress.* Reporters relay the results of this interaction by visual and linguistic metaphors to the human investigator. Artificial intelligence techniques (e.g. Organizer search strategies to achieve minimum "stress," natural language processing by the reporters, etc.) underly the entire system’s effectiveness as well as the functioning of its components. If desired, this metaphor can duplicate the results of least squares projection on a linear subspace. The range of analyses it makes possible though, is vastly broader and richer.

2.16.4 References


3. Education

3.1 Artificial Intelligence Education at Texas

The University of Texas at Austin has for many years been a leader in AI education. Courses in AI have been offered at UT for over 15 years. UT faculty have written numerous books on the subject, including Artificial Intelligence (Rich, McGraw-Hill, 1983), Computations From The English (Simmons, Prentice-Hall, 1983), and A Computational Logic (Boyer and Moore, Academic Press, 1979). In addition, extensive course notes have been developed for AI-related courses; for example, Novak has developed course notes on Lisp (120 pages) and graduate AI (450 pages). In addition to the usual classroom experience, several of our faculty also have experience in teaching specifically to professional audiences. The extensive experience and excellent written materials of UT faculty allow them to provide education in AI that is second to none.

The development of single-user Lisp machines promises to greatly increase the speed with which A.I. applications can be developed. At the same time, these machines present a new challenge for education. Because the mode of interaction between people and Lisp machines is so different from interaction with traditional timesharing machines -- especially in the use of high-resolution graphics and "mouse" pointing devices for human-machine interaction -- there is a need to educate students directly in the use of these machines. We have included in the budget funds for purchase of twenty Lisp machines to be used to support education; these machines are expected to be low-cost machines of the Xerox 1108 "Dandelion" class. Such machines provide the interaction needed for teaching, but cost much less than the faster machines needed for research.

3.2 Courses in Artificial Intelligence

In this section, we describe the existing educational program in Artificial Intelligence at the University of Texas at Austin. Courses can be added in conjunction with creation of the proposed Artificial Intelligence Project; in particular, the University allows graduate "topics" courses to be offered if there are at least five students registered. Given the broad coverage of the field of AI by the faculty at UT, it is likely that any special courses desired by the Army can be covered using this mechanism.

The existing AI-related courses offered at the University of Texas are described below. Some courses are offered only once per year or once per two years, as demand warrants.

3.2.1 Computer Sciences

CS 135 Introduction to Lisp. Introduction to Lisp programming for those who have experience in another language. Programming assignments are solved on the computer.

CS 343 Artificial Intelligence. (Undergraduate) Basic notions of Artificial
Intelligence. The use of computers to perform such tasks as Problem solving, Game-playing, theorem-proving, Pattern recognition, etc. Discussion of methods of search, representation, learning, etc.

**CS 381K Artificial Intelligence.** (Graduate) Use of computers in problem solving, game-playing, theorem-proving, pattern recognition, and related tasks; discussion of methods of search, representations, learning, etc.

**CS 388 Computational Linguistics.** Methods and procedures for syntactic and semantic analysis of natural language text, question answering, and discourse representation.

**CS 395 Procedural Logic for Problem Solving.** Logical representation of problems as systems of axioms; procedural logic for exploring search spaces, managing the frame problem and achieving reason maintenance system. Projects in PROLOG or LISP.

**CS 395T Human-Machine Problem Solving.** Techniques for enhancing person-machine problem solving, including language design (command and query languages), applications of menus and graphics, intelligent help facilities, and computer-aided instruction.

**CS 395T Artificial Intelligence Problem Solving.** Overview of various problem solving and planning systems. Study of knowledge representation techniques. Investigation of methodologies for integrating knowledge into problem solving search procedures.

**CS 395T Computational Logic.** The use of logic to reason mechanically about programs. A logic supported by a theorem prover will be presented. Particular emphasis on inductive proofs of recursive programs.

**CS 395T Expert Systems.** Case studies of existing expert systems in medical diagnosis, minerals prospecting, mass spectrographic interpretation. Tools for building expert systems. An expert system is written by each student as a term project, using the EMYCIN system.

**CS 395T Formal Semantics and Verification.** Sequential execution: partial and total correctness, deductive semantics, operational semantics, denotational semantics, formal derivation of programs; parallel execution: partial correctness, deadlock and starvation, methodology, parallel vs. distributed execution.

**CS 395T Text-Knowledge Systems.** Methods for representing natural language texts and questions as logical propositions. The use of procedural logic for answering questions and generating summaries.

**CS 395T Automatic Theorem Proving.** Proving theorems by computer. A survey of
various methods used in Automated theorem proving: Resolution, Natural Deduction, Inequality packages, complete sets of reductions, rewrite rules, Horn-clause provers, etc. Writing programs to do some of these.

3.2.2 Electrical Engineering

EE 381K *Digital Image Processing*.

3.2.3 Mechanical Engineering

ME 392M *Robotic Manipulators*. Kinematic, dynamic, control, actuation, sensing, and programming issues concerning robotic manipulators are examined.

3.2.4 Philosophy


PHL 358 *Topics in Philosophical Logic*. Survey of uses and extensions of symbolic logic, including such topics as modal logic, epistemic and deontic logics, higher-order logics.

PHL 359 *Topics in Symbolic Logic*. A theoretical study of logic, including axiomatization, definition, consistency and completeness; formal syntax.

PHL 374K *Logical Theory*. Topics vary and have included constructivist logic, systems of logic, and advanced symbolic logic.

PHL 382 *Philosophy of Language*. Seminar with variable topics including: model theoretic analysis of the meaning of expressions and constructions of natural language; conversational implicature and pragmatic aspects of communication; situation semantics and discourse representation structure theories.

PHL 383 *Perception*. Epistemological and physiological theories of perception and their relationships to computational theories of perception.

PHL 386 *Philosophy of Science*. Seminar with variable topics including: Fundamental problems in the philosophy of the natural sciences; laws and explanations; causation; unity of science; social laws and social structures.

PHL 389 *Logic*. Seminar with variable topics including: survey of mathematical logic; deviant logics; advanced set theory.
3.2.5 Linguistics

LING 340 *Automata theory.* This course covers the principal results in the theory of automata and their relations to formal grammars. There will be a review of set theory and logic. Other topics include finite automata and regular expressions, context-free languages and push-down automata, Turing machines, Church's thesis, uncomputability and computational complexity.

LING 372M *Logical Foundations of Linguistics.* Introduction to those aspects of finite mathematics that are required for advanced work in modern linguistic approaches to natural language. Includes topics in elementary set theory and logic (propositional predicate calculus).

LING 393 *Computer methods for linguists.* Introductory course for people with little or no background with computers. The uses of computers in research on natural language, and the use of natural language in computational systems (e.g., systems enabling people to use natural language for interacting with computers). Class project, to design and program a small question-answering system for English. The programming language is Lisp and students will learn something about Lisp as part of the course.

3.2.6 Psychology

PSY 387N *Fundamentals of Perception.* Overview of theory and research in the areas of visual perception and perceptual information processing.


3.3 Fellowships

One of the goals of the establishment of the Artificial Intelligence Project is to increase the number of students who are trained in this area. There is a severe shortage of people who have been trained in AI, and this shortage of personnel is limiting the application of AI techniques to defense problems. The proposed program will provide research assistantships to graduate students. However, there is a need to support students during the earlier phases of their education, before they are qualified to serve as research assistants. Without such support, promising students may be drawn off into other areas of specialization.

We propose that the Army support a number of fellowships in AI and AI-related areas in order to attract students for training in these areas. We propose that fellowships be supported at the level of 10 Ph.D., 10 Master's, and 10 undergraduate fellowships per year. The students who receive these fellowships will be carefully chosen by the Executive Committee of the Project. Ph.D. and Master's fellowship holders will each receive a fellowship for their first year of study; it is expected that this year will be spent
primarily on classwork. During the first year, students will be encouraged to associate themselves with one of the ongoing research projects, with the expectation that they will become research assistants on these projects following their first year of study. Undergraduate fellowships will be granted to senior students to encourage them to study AI and encourage them to pursue graduate study in the field.

3.4 Education for Army Personnel

A primary goal of the Army in supporting universities in the Artificial Intelligence area is to provide education and training in AI for Army personnel. The University of Texas at Austin is especially well-equipped to provide education in AI to Army personnel in appropriate ways and at appropriate levels of specialization. This section describes the educational services that we propose to provide to the Army.

3.4.1 Annual Conference

We propose that the University of Texas hold an annual conference at which the results of the preceding year’s research and education projects will be discussed. This conference will serve several purposes. First, it will serve to inform the Army about the research work that is being sponsored; by holding the conference in Austin, it will be possible for Army personnel to hold more detailed discussions with investigators of projects that are of special interest to them. A second purpose that will be served by the conference is that it will provide a forum for the Army to inform the members of the Project about Army needs. This may suggest new research projects of mutual interest. In addition, the conference will allow the Army and the Project members to discuss the educational program and its needs. We have included funds for holding the proposed Annual Conference in the budget.

3.4.2 Graduate Training for Army Personnel

The Army can send graduate students to the University of Texas at Austin for regular graduate training; presumably, such students will be able to work with members of the Project on research projects. Of course, such students will be expected to gain admission to the University through the normal procedure. The breadth and depth of the AI education offered at UT, as described in previous sections, makes UT an excellent choice for educating Army personnel. The equipment requested in this proposal will also serve to educate Army personnel in the use of the latest AI hardware and software tools.

3.4.3 Summer Short Courses

Summer 1-week or 2-week courses will be offered to meet Army needs for overview training. Examples of course topics include:

- Management Overview of Artificial Intelligence
- Technical Introduction to Artificial Intelligence
- Introduction to Lisp
- Advanced Lisp
- Introduction to PROLOG
- Problem Solving Techniques
- Expert Systems
- Natural Language Processing
- Semantic Representation and Data Structures
- Probabilistic Theories of Meaning
- Logics of Partial Information
- Syntax and Parsing

The University of Texas has very broad faculty coverage in the various areas of Artificial Intelligence. Working with the Army to determine those courses that would be of most interest to Army personnel, we will develop and offer additional summer short courses to meet Army needs.

3.4.4 Summer Institutes

In cooperation with the Army, we will offer Summer Institutes for appropriately qualified Army personnel. Typically, these will involve some formal course work (either some of the two-week Summer Short Courses or some regular A.I. courses offered during the Summer Session) coupled with project work on projects assigned and supervised by faculty Principal Investigators of the project and using the proposed educational equipment. Examples of Summer Institutes that might be offered include:

1. Lisp Programming.
3. Natural Language Processing.

3.4.5 On-Site Courses and Consulting

In some cases, it will be desirable from the Army's viewpoint to have short course offerings at Army facilities. Faculty from this Project will be available to offer short courses at locations to be designated by the Army on topics of interest to the Army; the specific topics to be covered will be decided in consultation with the Army. We have not included travel expenses for such visits in this proposal, since such expenses are unknown at this time; instead, we are assuming that the Army will furnish travel for such visits.
Consulting by faculty members is a valuable service that can be furnished to the Army. Since Army needs and individual specialties vary, arrangements for consulting, in accord with all applicable University, State, and Federal regulations, will be made between each individual involved and the agency that desires the consulting services.
4. Management

Accomplishment of the objectives of this proposal requires a strong management plan. We have developed such a plan, and have included in the proposed budget sufficient administrative staff to insure that the objectives are met. The Artificial Intelligence Project will be located in the Department of Computer Sciences, and will be under the general supervision of the Department. General policy of the Project will be determined by an Executive Committee composed of the faculty members who are Principal Investigators of the research projects; new Principal Investigators and new research projects will added in the future by vote of the Executive Committee, and with concurrence of the Army.

The Artificial Intelligence Project will be managed by a Director, elected annually by the Executive Committee from among its membership. The Director for the first year will be Gordon S. Novak Jr. The Director will be assisted by a staff Administrator and secretary. A Computer Facilities Manager will be hired to maintain software on the Lisp machines used by the Project and on related machines. Two additional secretaries will be hired to aid in preparation of reports, technical papers, and classroom materials.

Each research project will be managed by the faculty Principal Investigator, under general supervision by the Project Director and the Executive Committee. Progress reports will be produced annually for each project (or more frequently if desired by ARO); in addition, each project will be reviewed at the annual Review Conference with Army personnel. Proposals for each year's project work will be submitted annually to the Army for review and approval; this will allow projects to be added as new faculty are hired, and will allow existing projects to be modified as needed to follow promising research directions. All projects will be managed in accordance with the policies and procedures of the University and applicable Federal regulations.
5. Equipment

The University and the Department already have a solid base of research equipment; it is not necessary to start from scratch, but only to supplement the existing equipment. The main research equipment that is proposed is addition of Lisp machines for use by the individual research projects. The value of these machines for research, and their cost-effectiveness, has been amply demonstrated. Acquisition of these powerful tools will allow us to get maximum productivity from scarce AI researchers.

We propose that Lisp machines be acquired initially at the rate of one machine per research project, and thereafter at a rate of one-third of that amount per year. For planning purposes, we are assuming a Symbolics 3600 as the basic Lisp machine to be acquired; at the time of actual equipment acquisition, we will carefully study the available machines and choose the most useful and cost-effective machines for our needs. We expect the Lisp machine market to be competitive, so that other machines may be more cost-effective in the future.

Funds are included in the budget for maintenance of the equipment to be acquired. We have assumed standard contract maintenance rates; at some point, it may become more cost-effective for us to hire someone to maintain the machines.

The budget includes funds to hire a Computer Facilities Manager for support of the Lisp machines. Experience has shown that these are complex machines, and that experienced personnel are required to maintain current versions of software, interface with software on remote machines used as file servers and print servers, diagnose and report software problems, etc.