

SUMEX
STANFORD UNIVERSITY
MEDICAL EXPERIMENTAL COMPUTER RESOURCE
RR-00785

ANNUAL REPORT – YEAR 16

Submitted to

BIOMEDICAL RESEARCH TECHNOLOGY PROGRAM
NATIONAL INSTITUTES OF HEALTH

June 1, 1989

STANFORD UNIVERSITY SCHOOL OF MEDICINE

Edward H. Shortliffe, Principal Investigator

Edward A. Feigenbaum, Co-Principal Investigator

DEPARTMENT OF HEALTH AND HUMAN SERVICES
PUBLIC HEALTH SERVICE
NATIONAL INSTITUTES OF HEALTH

DIVISION OF RESEARCH RESOURCES
BIOMEDICAL RESEARCH TECHNOLOGY PROGRAM

ANNUAL PROGRESS REPORT
PART I., TITLE PAGE

1. PHS GRANT NUMBER: 5 P41 RR00785-16
2. TITLE OF GRANT: SUMEX — Stanford University
Medical Experimental Computer
Resource
3. NAME OF RECIPIENT INSTITUTION: Stanford University
4. HEALTH PROFESSIONAL SCHOOL: School of Medicine
5. REPORTING PERIOD:
5a. FROM: 08-01-88
5b. TO: 07-31-89
6. PRINCIPAL INVESTIGATOR:
6a. NAME: Edward H. Shortliffe, M.D., Ph.D.
6b. TITLE: Associate Professor of Medicine
and Computer Science
- 6c. SIGNATURE: 

7. DATE SIGNED: June 1, 1989
8. TELEPHONE: 415-723-6979

Table of Contents:

I. Title Page	1
II. Description of Program Activities	3
II.A. Scientific Subprojects	3
II.B. Books, Papers, and Abstracts	3
II.C. Resource Summary Table	3
III. Narrative Description.....	5
III.A. Summary of Research Progress	5
III.A.1 Resource Overview	5
III.A.1.1 SUMEX-AIM as a Resource	5
III.A.1.2 Significance and Impact in Biomedicine	11
III.A.1.3 Summary of Current Resource Goals	12
III.A.2 Details of Technical Progress.....	16
III.A.2.1 Key Areas of Progress.....	16
III.A.2.3. Core ONCOCIN Research	26
(1) Overview of the ONCOCIN Therapy Planning System	26
(2) Implementation of the ONCOCIN Workstation in the Stanford Clinic	27
(3) E-ONCOCIN: Domain Independent Therapy Planning	28
(4) OPAL: Graphical Knowledge Acquisition Interface.....	29
(5) Generalized Knowledge Acquisition through PROTÉGÉ	31
(6) Speech Input to Expert Systems	31
(6.1) Prototype Speech Hardware/Software System	31
(6.2) Speech Experiments	33
(7) Object Language Support for ONCOCIN Project.....	34
(8) Personnel	34
III.A.2.2. Core AI Research	36
(1) Rationale.....	36
(2) Highlights of Progress.....	37
(2.1) Large Multi-use Knowledge Bases for Science and Engineering	37
(2.2) Adaptive Intelligent Systems	40
(2.3) Advanced Architectures.....	41
(2.4) Knowledge Acquisition and Machine Learning	45
(2.5) Symbolic Simulation.....	45
III.A.2.4. Core System Research and Development.....	47
(1) Introduction and Overview	47

(2) The Phase-Out of the DECSys-20.....	49
(3) The New SUN-Based SUMEX-AIM Resource.....	51
(3.1) File Access and Management.....	52
(3.2) The SUMEX Perpetual Archive System.....	53
(3.3) Printing Services.....	55
(4) Electronic Mail.....	56
(4.1) Macintosh client - MacMM.....	58
(4.2) Mail Reader User-Interface.....	58
(4.3) The Mail Composition User Interface.....	60
(4.4) Texas Instruments Explorer Client.....	60
(4.5) DEC-20 IMAP2 Server.....	60
(4.6) UNIX IMAP2 Server.....	61
(4.7) Transition Strategy and Plan.....	62
(5) Lisp Systems.....	62
(5.1) Standards.....	62
(5.2) Lisp System Performance.....	63
(5.3) Lisp Programming Environments.....	65
(6) Workstation System Environments.....	66
(6.1) Macintosh II Workstations.....	66
(6.2) Texas Instruments Explorers.....	68
(6.3) SUN Workstations.....	72
(6.4) NeXT Workstations.....	73
(6.5) Xerox D-Machines.....	75
(6.6) Symbolics Lisp Machines.....	79
(6.7) HP 9836 Workstations.....	80
(7) Remote Workstation Access, Virtual Graphics, and Windows.....	80
(7.1) Remote Access.....	80
(7.2) Virtual Graphics and Windows.....	80
(7.3) Remote Graphics Applications.....	81
(8) Network Services.....	84
(8.1) National and Wide-Area Networks.....	84
(8.2) Local Area Networks - LAN's.....	86
(9) Distributed Information Resources and Access.....	88
(10) Distributed system operation and management.....	90
III.A.2.5. Relevant Core Research Publications.....	91
III.A.2.6. Resource Equipment.....	98
(1) Purchases This Past Year.....	98
(2) Current Subsystem Configurations.....	100

III.A.2.7. Training Activities	105
III.A.2.8. Resource Operations and Usage	108
(1) Operations and Support.....	108
(2) Resource Usage Details	108
(2.1) Overall Resource Loading Data	109
(2.2) Individual Project and Community Usage	110
III.B. Research Highlights	117
III.B.1. INTERNIST-I/QMR	118
III.B.2. PathFinder.....	119
III.B.3. The Distributed SUMEX-AIM Community	120
III.B.4. ONCOCIN.....	121
III.C. Administrative Changes.....	123
III.D. Resource Management and Allocation	124
III.D.1. Overall Management Plan.....	124
III.D.2. Cost Center.....	124
III.E. Dissemination of Resource Information	127
III.E.1. Software Distribution.....	127
III.E.2. AIM Community Systems Support.....	128
III.E.3. Video Tapes and Films.....	128
III.E.4. Special Seminars.....	128
III.F. Suggestions and Comments	129
III.F.1. Resource Organization	129
III.F.2. Electronic Communications	129
IV. Description of Scientific Subprojects.....	130
IV.A. Stanford Projects.....	131
IV.A.1. Guardian Project.....	132
IV.A.2. MOLGEN Project.....	137
IV.A.3. ONCOCIN Project.....	143
IV.A.4. PENGUIN Project.....	157
IV.A.5. PROTEAN Project	166
IV.A.6. Reasoning Under Uncertainty	174
IV.A.7. VentPlan Project.....	183
IV.B. National AIM Projects	190
IV.B.1. INTERNIST-I/QMR Project	191
IV.B.2. MENTOR Project.....	197

IV.C. Pilot Stanford Projects.....	202
IV.C.1. REFEREE Project.....	203
IV.D. Pilot AIM Projects.....	210
IV.D.1. The Pathfinder Project.....	211
Appendix A: Knowledge Systems Laboratory Brochure.....	219
Appendix B: Lisp Performance Studies.....	229
Appendix C: AIM Management Committee Membership.....	261

List of Figures:

Figure 1. NSFNet Configuration as of January 1989.....	85
Figure 2. SUMEX-AIM DEC 2060 Configuration.....	100
Figure 3. SUMEX-AIM SUN-4 Configuration.....	101
Figure 4. SUMEX-AIM SUN-3 File Server Configuration.....	101
Figure 5. SUMEX-AIM Xerox File Server Configuration.....	102
Figure 6. SUMEX-AIM VAX File Server Configuration.....	103
Figure 7. SUMEX-AIM Develcon X.25/TCP-IP Gateway Configuration.....	103
Figure 8. SUMEX-AIM Ethernet Configuration.....	104
Figure 9. Total CPU Hours Consumed by Month.....	110
Figure 10. CPU Usage Histogram by Project and Community.....	111
Figure 11. Table of Resource Use by Project.....	112

II. Description of Program Activities

This section corresponds to the predefined forms required by the Division of Research Resources to provide information about our resource activities for their computerized retrieval system. These forms have been submitted separately and are not reproduced here to avoid redundancy with the more extensive narrative information about our resource and progress provided in this report.

II.A. Scientific Subprojects

Our core research and development activities are described starting in section III.A.2, our training activities are summarized in section III.A.2.7, and the progress of our collaborating projects is detailed starting in section IV.

II.B. Books, Papers, and Abstracts

The list of recent publications for our core research and development work is given in section III.A.2.5 and those for the collaborating projects are in the individual reports starting in section IV.

II.C. Resource Summary Table

The details of resource usage, including a breakdown by the various subprojects, is given in the tables starting in section III.A.2.8

III. Narrative Description

III.A. Summary of Research Progress

III.A.1 Resource Overview

This is an annual report for year 16 of the SUMEX-AIM resource (grant RR-00785), the third year of a 5-year renewal period to support further research on applications of artificial intelligence in biomedicine. For the technical and administrative reasons discussed in earlier reports, the SUMEX project now includes the continuation of work on the development and dissemination of medical consultation systems (ONCOCIN) that had been supported before 1986 as resource-related research under grant RR-01631. Progress on core ONCOCIN research is therefore now reported here as well.

The originally proposed research program (June 1985 renewal application) included an ambitious plan to:

- Continue our long-range core research efforts on knowledge-based systems, aimed at developing new concepts and methodologies needed for biomedical applications.
- Substantially extend ONCOCIN research on developing and disseminating clinical decision support systems.
- Develop the core systems technology to move the national SUMEX-AIM community from a dependence on the central SUMEX DEC 2060 to a fully distributed, workstation-based computing environment.
- Introduce these systems technologies into the SUMEX-AIM community with appropriate communications and managerial assistance to responsibly phase out the central resource and DEC 2060 mainframe in a manner that will support community efforts to become self-sustaining and to continue scientific interactions through fully distributed means.
- Maintain our aggressive efforts at training and dissemination to help exploit the research potential of this field.

III.A.1.1 SUMEX-AIM as a Resource

SUMEX and the AIM Community

Since the SUMEX-AIM resource was established in late 1973, computing technology and biomedical artificial intelligence research have undergone a remarkable evolution and SUMEX has both influenced and responded to these changing technologies. It is widely recognized that our resource has fostered highly influential work in biomedical AI — work from which much of the expert systems field emerged — and that it has simultaneously helped define the technological base of applied AI research.

The focus of the SUMEX-AIM resource continues to emphasize research on artificial intelligence techniques that guide the design of computer programs

that can help with the acquisition, representation, management, and utilization of the many forms of medical knowledge in diverse biomedical research and clinical care settings — ranging from biomolecular structure determination and analysis, to molecular biology, to clinical decision support, to medical education. Nevertheless, we have long recognized that the ultimate impact of this work in biomedicine will be realized through its synthesis with the full range of methodologies of medical informatics, such as data bases, biostatistics, human-computer interfaces, complex instrument control, and modeling. From the start, SUMEX-AIM work has been grounded in real-world applications, like systems for the interpretation of mass spectral information about biomolecular structures, chemical synthesis, interpretation of x-ray diffraction data on crystals, cognitive modeling, infectious disease diagnosis and therapy, DNA sequence analysis, experiment planning and interpretation in molecular biology, and medical instruction. Our current work extends this emphasis in application domains such as oncology protocol management, clinical decision support, protein structure analysis, and data base information retrieval and analysis. All of these research efforts have demanded close collaborations with diverse parts of the biomedical research community and the integration of many computational methods from those domains with knowledge-based approaches. Even though in the beginning the "AI-in-medicine" community was quite small, it is perforce no longer limited and easily-defined, but rather is spreading and is inextricably linked with the many biomedical applications communities we have collaborated with over the years. Driven both by the on-going diffusion of AI and by the development of personal computer workstations that signal the practical decentralization of computing resources, we must develop new resource communication and distributed computing technologies that will continue to facilitate wider intra- and inter-community communication, collaboration, and sharing of biomedical information.

The SUMEX Project has demonstrated that it is possible to operate a computing research resource with a national charter and that the services providable over networks were those that facilitate the growth of AI-in-Medicine. SUMEX now has a reputation as a model national resource, pulling together the best available interactive computing technology, software, and computer communications in the service of a national scientific community. Planning groups for national facilities in cognitive science, computer science, and biomathematical modeling have discussed and studied the SUMEX model and new resources, like the BIONET resource for molecular biologists, are closely patterned after the SUMEX example.

The projects SUMEX supports have generally required substantial computing resources with excellent interaction. Today, with the dramatic explosion of high-performance workstations that are more and more generally available, the need for a central source of raw computing cycles has significantly diminished. In place of being a distributor of CPU cycles, SUMEX has become a communications cross-roads and a source of AI and computer systems software and expertise.

SUMEX has demonstrated that a computer resource is a useful "linking mechanism" for bringing together electronically teams of experts from different disciplines who share a common problem focus. AI concepts and software are among the most complex products of computer science. Historically it has not been easy for scientists in other fields to gain access to and mastery of them. Yet the collaborative outreach and dissemination efforts of SUMEX have been able to bridge the gap in numerous cases. Over 40 biomedical AI application projects have developed in our national community and have been supported directly by SUMEX computing resources over the years — many more have benefitted indirectly through access to the software, information, and advice offered by the SUMEX resource.

The integration of AI ideas with other parts of medical informatics and their dissemination into biomedicine is happening largely because of the development in the 1970's and early 1980's of methods and tools for the application of AI concepts to difficult professional-level problem solving. Their impact was heightened because of the demonstration in various areas of medicine and other life sciences that these methods and tools really work. Here SUMEX has played a key role, so much so that it is regarded as "the home of applied AI."

SUMEX has been the home of such well-known AI systems as DENDRAL (chemical structure elucidation), MYCIN (infectious disease diagnosis and therapy), INTERNIST (differential diagnosis), ACT (human memory organization), MOLGEN/BIONET (tools for DNA sequence analysis and molecular biology experiment planning), ONCOCIN (cancer chemotherapy protocol advice), SECS (chemical synthesis), EMYCIN (rule-based expert system tool), and AGE (blackboard-based expert system tool). Since 1980, our community has published a fifteen books that give a scholarly perspective on the scientific experiments we have been performing. These volumes, and other work done at SUMEX, have played a seminal role in structuring modern AI paradigms and methodology.

The Future of SUMEX-AIM

Given this background, what is the future need and course for SUMEX as a resource — especially in view of the on-going revolution in computer technology and costs and the emergence of powerful single-user workstations and local area networking? The answers remain clear.

Basic Research on AI in Biomedicine

At the deepest research level, despite our considerable success in working on medical and biological applications, the problems we can attack are still sharply limited. Our current ideas fall short in many ways against today's important health care and biomedical research problems brought on by the explosion in medical knowledge and for which AI should be of assistance. Just as the research work of the 70's and 80's in the SUMEX-AIM community

fuels the current practical and commercial applications, our work of the late 80's will be the basis for the next decade's systems.

The report of the panel on medical informatics¹, convened late in 1985 by the National Library of Medicine to review and recommend twenty-year goals for the NLM, listed among its highest priority recommendations the need to greatly expand and aggressively pursue an interdisciplinary research program to develop computational methods for acquiring, representing, managing, and using biomedical knowledge of all sorts for health care and biomedical research. Similar recommendations have been stated recently by the panel on Information Technology and the Conduct of Research of the National Academy of Science². These are precisely the problems which the SUMEX-AIM community has been working on so successfully and which will require work well beyond the five year funding period we have requested. It is essential that this line of research in the SUMEX-AIM community, represented by our core AI research, the ONCOCIN research, and our collaborative research groups, be continued.

The Changing Role of the Central Resource

At the resource level, there are changing, but still growing, needs for computing resources for the active AIM research community to continue its work over the next five years. The workstations to which we directed our attention in 1980 have now demonstrated their practicality as research tools and, increasingly, as mechanisms for disseminating AI systems as cost-effective decision aids in clinical settings such as private offices. The era of highly centralized general machines for AI research is nearly at an end and is being replaced by networks of distributed but heterogeneous single-user machines sharing common information resources and communication paths among members of the biomedical research community.

Most of our community groups have been able to take advantage of local computing facilities, with SUMEX-AIM providing a central cross-roads for communications and the sharing of programs and knowledge. In its core research and development role, SUMEX-AIM has its sights set on the hardware and software systems of the next decade. We expect major changes in the distributed computing environments that are just now emerging in order to make effective use of their power and to adapt them to the development and dissemination of biomedical AI systems for professional user communities. In its training role, SUMEX is a crucial resource for the education of badly needed new researchers and professionals to continue the development of the biomedical AI field. The "critical mass" of the existing

¹ *Long Range Plan*. Report of the Board of Regents, National Library of Medicine. National Institutes of Health. January 1987.

² *Information Technology and the Conduct of Research — The User's View*. Report of the Panel on Information Technology and the Conduct of Research, National Academy of Sciences. National Academy Press. 1989.

physical SUMEX resource, its development staff, and its intellectual ties with the Stanford Knowledge Systems Laboratory (KSL — see Appendix A for a summary of current KSL research activities), make this an ideal setting to integrate, experiment with, and export these methodologies for the rest of the AIM community.

We will continue our experimental approach to distributed systems, learning to build and exploit distributed networks of these machines and to build and manage graceful software for these systems. Since decentralization is central to our future, we must learn its technical characteristics.

Resource Sharing

An equally important function of the SUMEX-AIM resource is an exploration of the use of computer communications as a means for interactions and sharing between geographically remote research groups engaged in biomedical computer science research and for the dissemination of AI technology. This facet of scientific interaction is becoming increasingly important with the explosion of complex information sources and the regional specialization of groups and facilities that might be shared by remote researchers¹. Another of the key recommendations of the NLM medical informatics planning panel² was that high-speed network communication links be established throughout the biomedical research community so that knowledge and information can be shared across diverse research groups and that the required interdisciplinary collaborations can take place. Recent efforts to establish a national NSFNet³, largely to support the supercomputer projects funded by NSF but also to replace and upgrade part of the national research community linkage that the now aging ARPANET has supported, have made important progress. Still, these efforts do not encompass the broad range of biomedical research groups that need national network access and to date, the NIH has not played an aggressive role in the interagency Research Internet coordination efforts. We must work to build a stronger institutional support for a National Research Network.

SUMEX continues to be an important pathfinder to develop the technology and community interaction tools needed to expand community system and communication resources. Our community building effort is based upon the developing state of distributed computing and communications technology

¹ Lederberg, J. "Digital Communications and the Conduct of Science: the New Literacy." *Proc. IEEE*, 66(11):1314-1319, 1978.

Coulter, C. L. "Research Instrument Sharing." *Science*, 201(4354), 1978.

Newell, A., and Sproull, R. F. "Computer Networks - Prospects for Scientists." *Science*, 215(4534):843, 1982.

² *NLM Long Range Plan: Medical Informatics*. NLM Planning Panel 4. National Library of Medicine, National Institutes of Health. January 1987.

³ Marshal, E. "NSF Opens High-Speed Computer Network." *Science*. 243: 22-23, 1989.

and we have therefore turned our core systems research to actively supporting the development of distributed computing and communications resources to facilitate collaborative project research and continued inter-group communications.

Summary of Long-term Goals

- Maintain the synergistic relationship between SUMEX core system development, core AI research, our experimental efforts at disseminating clinical decision-making aids, and new applications efforts.
- Continue to serve the national AIM research community, less and less as a source of raw computing cycles and more and more as a transfer point for new technologies important for community research and communication. We will also continue our coordinating role within the community through electronic media and periodic AIM workshops
- Maintain our connections to national networks (e.g., NSFNet, ARPANET, and TELENET) and our local Ethernet and assist other community members to establish similar links by example, by integrating and providing enabling software, and by offering advice and support within our resources.
- Focus new computing resource developments on more effective exploitation of distributed workstations through better communication and cooperative computing tools, using transparent digital networking schemes.
- Enhance the computing environments of workstations so that only minimal dependency on central, general-purpose computing hosts remains and these mainframe time-sharing systems can be phased out eventually. Remaining central resources will include servers for communications, community information resources, and special computing architectures (e.g., shared- or distributed-memory symbolic multiprocessors) justified by cost-effectiveness and unique functionality.
- Incrementally phase in, disseminate, and evaluate those aspects of the local distributed computing resource that are necessary for continuing national AIM community support within this distributed paradigm. This will ultimately point the way towards the distributed computing resource model that we believe will interlink this community well into the next decade.
- Responsibly phase out the existing DEC 2060 machine as effective distributed computing alternatives become widely available. Because of severe budget pressures, the 2060 was taken out of routine service during this past year in a much more accelerated fashion than was planned or was comfortable for AIM users to acclimate to the new UNIX operating system environment. We are still finishing up a number of interim systems alternatives to discontinued 2060 services not available in standard UNIX environments.

- Continue the central staff and management structure, essentially unchanged in function during the five-year transition period, except for the merging of the core part of the ONCOCIN research with the SUMEX resource.

III.A.1.2 Significance and Impact in Biomedicine

Artificial intelligence is the computer science of representations of symbolic knowledge and its use in symbolic inference and problem-solving processes. Projects in the SUMEX-AIM community are concerned in some way with the application of AI to biomedical research and the resource has given strong impetus and support to knowledge-based system research in biomedicine. For computer applications in medicine and biology, this research path is crucial. Medicine and biology are not presently mathematically-based sciences; unlike physics and engineering, they are seldom capable of exploiting the mathematical characteristics of computation. They are essentially inferential, not calculational, sciences. If the computer revolution is to affect biomedical scientists, computers will be used as inferential aids.

The growth in medical knowledge has far surpassed the ability of a single practitioner to master it all, and the computer's superior information processing capacity thereby offers a natural appeal. Furthermore, the reasoning processes of medical experts are poorly understood; attempts to model expert decision-making necessarily require a degree of introspection and a structured experimentation that may, in turn, improve the quality of the physician's own clinical decisions, making them more reproducible and defensible. New insights that result may also allow us more adequately to teach medical students and house staff the techniques for reaching good decisions, rather than merely to offer a collection of facts which they must independently learn to utilize coherently.

Perhaps the larger impact on medicine and biology will be the exposure and refinement of the hitherto largely private heuristic knowledge of the experts of the various fields studied. The ethic of science that calls for the public exposure and criticism of knowledge has traditionally been flawed for want of a methodology to evoke and give form to the heuristic knowledge of scientists. AI methodology is beginning to fill that need. Heuristic knowledge can be elicited, studied, critiqued by peers, and taught to students.

The importance of AI research and its applications is increasing in general, without regard for the specific areas of biomedical interest. AI has been one of the principal fronts along which university computer science groups are expanding. The pressure from student career-line choices is great. Federal and industrial support for AI research and applications is vigorous, although support specifically for biomedical applications continues to be limited. All of the major computer manufacturers (e.g., IBM, DEC, TI, UNISYS, HP, Apple, and others) are using and marketing AI technology aggressively and many software companies are putting more and more products on the market. Many other parts of industry are also actively pursuing AI applications in

their own contexts, including defense and aerospace companies, manufacturing companies, financial companies, and others¹.

Despite the limited research funding available, there is also an explosion of interest in medical AI. The American Association for Artificial Intelligence (AAAI), the principal scientific membership organization for the AI field, has over 7000 members, several thousand of whom are members of the medical special interest group known as the AAAI-M. Speakers on medical AI are prominently featured at professional medical meetings, such as the American College of Pathology and American College of Physicians meetings; a decade ago, the words *artificial intelligence* were never heard at such conferences. And at medical computing meetings, such as the annual Symposium on Computer Applications in Medical Care (SCAMC) and the international MEDINFO conferences, the growing interest in AI and the rapid increase in papers on AI and expert systems are further testimony to the impact that the field is having.

AI is beginning to have a similar effect on medical education. Such diverse organizations as the National Library of Medicine, the American College of Physicians, the Association of American Medical Colleges, and the Medical Library Association have all called for sweeping changes in medical education, increased educational use of computing technology, enhanced research in medical computer science, and career development for people working at the interface between medicine and computing. They all cite evolving computing technology and (SUMEX-AIM) AI research as key motivators. At Stanford, we have a vigorous special graduate program in Medical Information Sciences for student training and research in AI. This program has many more applicants than available slots. Demand for these graduates, in both academic and industrial settings, is so high that students typically begin to receive solicitations one or two years before completing their degrees.

III.A.1.3 Summary of Current Resource Goals

The following outlines the specific objectives of the SUMEX-AIM resource during the current five-year award period begun in August 1986. It provides an overall research plan for the resource and the backdrop against which specific progress is reported. Note that these objectives cover only the resource nucleus; objectives for individual collaborating projects are discussed in their respective reports in Section IV. Specific aims are broken into five categories: 1) Technological Research and Development, 2) Collaborative

¹ Feigenbaum, E. A., McCorduck, P., and Nii, H. P. *The Rise of the Expert Company: How Visionary Companies are Using Artificial Intelligence to Achieve Higher Productivity and Profits*. Times Books, New York, NY, 1988.

Winston, P. H., and Prendergast, K. A. *The AI Business: Commercial Uses of Artificial Intelligence*. The MIT Press, Cambridge, MA, 1984.

Research, 3) Service and Resource Operations, 4) Training and Education, and 5) Dissemination.

Technological Research and Development

- NIH-based SUMEX funding and computational support for core research is complementary to similar funding from other agencies (including DARPA, NASA, NSF, NLM, private foundations, and industry) and contributes to the long-standing interdisciplinary effort at Stanford in basic AI research and expert system design. We expect this work to provide the underpinnings for increasingly effective consultative programs in medicine and for more practical adaptations of this work within emerging microelectronic technologies. Specific aims include:
- Basic research on AI techniques applicable to biomedical problems. Over the next term we will emphasize work on very large multi-use knowledge bases, blackboard problem-solving frameworks and architectures, knowledge acquisition or learning, constraint satisfaction, and qualitative simulation.
- Investigate methodologies for disseminating application systems such as clinical decision-making advisors into user groups. This will include generalized systems for acquiring, representing and reasoning about complex treatment protocols such as are used in cancer chemotherapy and which might be used for clinical trials in other domains.
- Support community efforts to organize and generalize AI tools and architectures that have been developed in the context of individual application projects. This will include retrospective evaluations of systems like the AGE blackboard experiment and work on new systems such as BB1, CARE, EONCOCIN, EOPAL, Meta-ONYX, and architectures for concurrent symbolic computing. The objective is to evolve a body of software tools that can be used to more efficaciously build future knowledge-based systems and explore other biomedical AI applications.
- Develop more effective workstation systems to serve as the basis for research, biomedical application development, and dissemination. We seek to coordinate basic research, application work, and system development so that the AI software we develop for the next 5-10 years will be appropriate to the hardware and system software environments we expect to be practical by then. Our purchases of new hardware will be limited to experimentation with state-of-the-art workstations as they become available for our system developments.

Collaborative Research

- Encourage the exploration of new applications of AI to biomedical research and improve mechanisms for inter- and intra-group collaborations and communications. While AI is our defining theme, we may consider exceptional applications justified by some other unique feature of SUMEX-AIM essential for important biomedical research. We

will continue to exploit community expertise and sharing in software development.

- Minimize administrative barriers to the community-oriented goals of SUMEX-AIM and direct our resources toward purely scientific goals. We will retain the current user funding arrangements for projects working on SUMEX facilities. User projects will fund their own manpower and local needs; actively contribute their special expertise to the SUMEX-AIM community; and receive an allocation of system resources under the control of the AIM management committees. We will progressively charge core SUMEX-AIM operations costs to Stanford users as DRR support for the central system (initially a DEC 2060) is phased out. Fees to national users will be delayed as long as financially possible.
- Provide effective and geographically accessible communication facilities to the SUMEX-AIM community for remote collaborations, communications among distributed computing nodes, and experimental testing of AI programs. We will retain the current ARPANET and TELENET connections for at least the initial term and will actively explore other advantageous connections to new communications networks and to dedicated links.

Service and Resource Operations

SUMEX-AIM does not have the computing or manpower capacity to provide routine service to the large community of mature projects that has developed over the years. Rather, their computing needs are better met by the appropriate development of their own computing resources when justified. Thus, SUMEX-AIM has the primary focus of assisting new start-up or pilot projects in biomedical AI applications in addition to its core research in the setting of a sizable number of collaborative projects. We do offer continuing support, when appropriate, for projects through the lengthy process of obtaining funding to establish their own computing base.

Training and Education

- Provide documentation and assistance to interface users to resource facilities and systems.
- Exploit particular areas of expertise within the community for assisting in the development of pilot efforts in new application areas.
- Accept visitors in Stanford research groups within limits of manpower, space, and computing resources.
- Support the Medical Information Science and other student programs at Stanford to increase the number of research personnel available to work on biomedical AI applications.
- Support workshop activities including collaboration with other community groups on the AIM community workshop and with individual projects for

more specialized workshops covering specific research, application, or system dissemination topics.

Dissemination

While collaborating projects are responsible for the development and dissemination of their own AI systems and results, the SUMEX resource will work to provide community-wide support for dissemination efforts in areas such as:

- Encourage, contribute to, and support the on-going export of software systems and tools within the AIM community and for commercial development.
- Assist in the production of video tapes and films depicting aspects of AIM community research.
- Promote the publication of books, review papers, and basic research articles on all aspects of SUMEX-AIM research.

III.A.2 Details of Technical Progress

This section gives an overview of progress for the nucleus of the SUMEX-AIM resource. A more detailed discussion of our progress in specific areas and related plans for further work are presented beginning in section III.A.2.2. Objectives and progress for individual collaborating projects are discussed in their respective reports in section IV. These collaborative projects collectively provide much of the scientific basis for SUMEX as a resource and our role in assisting them has been a continuation of that evolved in the past. Collaborating projects are autonomous in their management and provide their own manpower and expertise for the development and dissemination of their AI programs.

III.A.2.1 Key Areas of Progress

In this section we summarize highlights of SUMEX-AIM resource activities over the past year (May 1988 - April 1989), focusing on the resource nucleus. We have made continued significant progress in all of our areas of core research, including the ONCOCIN research on dissemination of clinical trial management tools, basic AI research, and distributed systems development.

Core ONCOCIN Research

- Our work has proceeded well along three main lines of research: 1) ONCOCIN, the therapy planning program and its graphical interface; 2) OPAL, a graphical knowledge entry system for ONCOCIN; and 3) ONYX, a strategic planning program designed to give advice in complex therapy situations. Each of these research components has in turn split into two parts: continued development of the cancer therapy versions of the system, and a generalization of each of the components for use in other areas of medicine (the prefix "E-" is added to the program names for the generalized versions). In addition, we have continued development of a generalized knowledge acquisition tool, named PROTÉGÉ, designed to encode descriptions of clinical trials. The system was the Ph.D. thesis work of Mark Musen, (who joined our faculty this year). The output of PROTÉGÉ is an OPAL-like input system designed for a target clinical area such as hypertension.
- Based on the success of our earlier ONCOCIN work, strong interest has developed, from such diverse quarters as the National Cancer Institute and the Stanford Hospital, for developing a fully operational version of ONCOCIN that can be broadly used in oncology clinics outside our research laboratory. This past year, the Stanford Hospital started a program to assist in the transfer of innovative medical technology out of the laboratory to patient care, committing approximately \$750K per year to seed this effort. ONCOCIN was selected as one of 10 projects to be funded from a large group of competing proposals. This presents a dilemma for the project that is still unresolved, namely, how to maintain a cohesiveness between ongoing research work to extend the various parts of ONCOCIN and generalize it for applications to other domains and at

the same time, meet the operational needs of a widely disseminated practical system. Much thought has gone into this problem this past year, including issues such as which of the modern workstation alternatives to select (Lisp machine, IBM PC, Apple Macintosh, SUN or NeXT UNIX workstation, ...), what language to pick (C, Lisp, ...), and can the research and operational systems really be consistent versions of a single system? In order to understand the scope and practical issues involved, we have begun an experiment to port ONCOCIN to a TI microExplorer running inside of a Mac II during the last six months. We have completed the translation of the Ozone object-oriented system, the temporal network and most of the reasoner. We will next approach the design of the user interface, which must be rewritten anew, since the current interface depends heavily on the graphical capabilities of the Xerox workstations. We are also starting a study of the overall design and specification of an "integrated" oncologist's workstation, under NCI sponsorship, that will lead to an attempt to coordinate federal, academic, and industrial efforts to implement such a system.

- Our E-ONCOCIN research has concentrated on understanding how protocols in medicine vary across subspecialties. We are examining several application areas: the intensive care unit, insulin treatment for diabetes, hypertension protocols, and both standard and complex cancer treatment problems. The diagnosis and therapy selection for patients in the intensive care unit is a natural application area because it is based on changing data and the need to determine the response to therapy interventions. We also felt that the area of insulin treatment for diabetes would be a good area to explore. Like cancer chemotherapy, the treatments for diabetes continues over a long period of time and has been the area of intensive protocol development. Unlike cancer chemotherapy, the treatment plan must handle multiple treatments in one day and deemphasises the use of multiple drugs (although there are a variety of types of insulin). Our initial experiments have shown that many of the elements of the ONCOCIN design are sufficiently general for other application areas, but that some specific elements (particularly the representation of temporal events) will have to be redesigned or extended. Another extension is to modify the framework so that it can work with established data base tools instead of the hand-tailored data base currently in use. In this work, we must be able to describe the changing clinical context and event intervals that show up in many diverse application areas. An example of a new area that we are exploring is the treatment of AIDS patients on clinical protocols. AIDS patients do not always follow the type of strict temporal schedules (e.g., regular visits to outpatient clinics) seen with oncology patients. They have a chronic disease with acute exacerbations of opportunistic infections. Furthermore, the medication schedule is interrupted by frequent hospitalizations and confounded by taking drugs not on the protocol.

Together, these factors will require a much more flexible model of the temporal dimension of treatment planning.

- We continued development of the OPAL system for graphical knowledge acquisition to facilitate protocol definition and knowledge base entry for the ONCOCIN oncology application area. A major accomplishment of this last year was to experimentally combine the OPAL and ONCOCIN programs into one working program, and to completely enter knowledge from OPAL using both the high level tools and lower level rule editors, but without needing to make changes at the ONCOCIN side of the system. Our experiments with OPAL, and our intention to generalize OPAL use outside of oncology protocols, suggests that we reorganize the OPAL program to use a relational data base to store its knowledge. We continue to explore the appropriate avenue for the connection of our knowledge acquisition systems to data bases, and have concentrated on the SQL query language to a relational data base using the client-server model (e.g., the physical data base may exist on a different machine than the knowledge acquisition tool — transmitting the query and the response over the network).
- With the current uncertainties in what workstation environment to use for future work, we began to explore alternative platforms for developing the interface for OPAL-like systems. We have begun experiments using HyperCard on the Mac II and Interface Builder on the NeXT machine. In order to build experience with the each of these possible platforms, we have re-implemented portions of OPAL system, and are analyzing the results. It is particularly hard to determine the best platform since the NeXT machine software is still in a rudimentary stage, and HyperCard on the Mac II has significant limitations including small "card size" and the inability to display multiple cards simultaneously.

We continue to work on the integration of speech-recognition technology into the interface to ONCOCIN. The project uses a commercially available continuous speech recognition product and a prototype ONCOCIN adaptation. The system uses the location of the cursor on the screen to provide a context for choosing candidate grammars with which to attempt recognition of a user's utterance. The system dynamically re-orders the list of candidate recognition grammars based on the dialog history. Albeit with limitations on the legal grammars, it is now possible to carry on most of the ONCOCIN data acquisition steps using speech alone or speech plus pointing with the mouse. We are also exploring a second medical record-keeping task — the creation of portions of a progress note that describes in textual form the changes in the patients status from week to week. We have also mounted the CMU SPHINX speech understanding system on our NeXT machines and are comparing its performance against the SSI hardware-based system we have been using.

Core AI Research

- In the last year, research has progressed on several fundamental issues of AI. As in the past, our research methodology is experimental, concentrating on building and analyzing actual systems. We have continued to explore the design and use of very large, multi-use knowledge bases with the hypothesis that both the problems of brittleness and over-specialization in current knowledge-based systems can be overcome. Some of the key directions for this work include knowledge representation, knowledge compilation, knowledge justification, model-based reasoning, and case-based reasoning. During the past year we have been exploring a variety of representations and the systems which employ them, including CYC from MCC, CLASS from Schlumberger, and QPE from Univ. of Illinois. In the study of knowledge compilation techniques, we note that effective problem solving is not typically carried out at the level of first principles, but rather at the level of more compact, efficient forms of knowledge, compiled from experience with specific tasks. We are developing an integrated scheme for using "first principles" knowledge of the physical world for simulation. Given a description of the structure of a device in terms of its constituent objects and their relations, the system identifies applicable physical laws, processes, types of matter, etc. and produces a set of equations to describe the behavior of the device. The equation model is then analyzed using the method of causal ordering to produce a model that reveals the dependency relations among the parameters of the model.
- Research has also progressed on our study of blackboard frameworks, especially as they relate to adaptive intelligent systems. Important questions for this work include: how can we design flexible control structures for powerful problem solving programs? How can we use these structures effectively in many problem domains? How can we represent processes and reason about their behavior, and perform intelligent actions under real-time requirements? This past year, we have begun or continued work on five domain-independent BB1 modules: the Focus module (provides a dynamic focus of attention); the ReAct module (provides time-sensitive problem detection and response capabilities); the ICE module (provides reasoning from first principles to handle complex or unfamiliar problems); the TPlan module (provides time-sensitive planning of coherent courses of action); and the TDB module (provides a temporally organized database of observed, expected, and intended models of external entities, and associated temporal reasoning functions).
- We have built upon earlier results in our parallel symbolic computing architectures project, including the SIMPLE CAD (Computer Aided Design) system for hierarchical, multiple level specification of computer architectures and the CARE parameterized, multiprocessor array emulator (specified in SIMPLE's specification languages and running on SIMPLE's simulator). These systems are in use by several research

groups at Stanford and have been ported to several external sites, including NASA Ames Research Center. A videotaped tutorial was held in June, 1988, attended by representatives from industry and government, which described the CARE/SIMPLE system, as well as the LAMINA programming interface. The attendees received instruction in use of the system for making measurements of the performance of various simulated multiprocessor applications. Due to rapidly growing interest in the SIMPLE/CARE system, a major effort is now underway to port it to wider class of hardware platforms. The system is currently being reimplemented in Common Lisp and the X window system, with Sun workstation as the initial target. During the past year, the research effort associated with SIMPLE/CARE has largely focussed on investigations of communication protocols and techniques for monitoring concurrent object-based applications. In other areas of our parallel architectures work, we have studied the measured speed-up of two different expert system applications, ELINT (a system for interpreting electronic intelligence signals) and AIRTRAC (a system for identifying and tracking aircraft based on diverse radar data). Our preliminary conclusions are that for relatively simple and well-structured applications such as ELINT, two (or possibly more) orders of magnitude speedup via parallel execution are possible. However, for complex and ill-structured applications such as AIRTRAC Path Association, speedup over a well-tuned serial program by using parallel execution is probably limited, at best, to an order of magnitude. Experiments are continuing to verify this preliminary conclusion.

The machine learning work has focussed this past year on explanation-based generalization and chunking work in the SOAR framework and inductive rule learning. This area of research is winding down due to the departures of Profs. Buchanan and Rosenbloom. During the past year finishing students extended the RL induction program to learn incrementally, that is from small sets of examples presented in sequence without benefit of looking at them all together. A front-end program was written to assist in the definition of RL's starting knowledge, the so-called "half-order theory". In our SOAR research, we completed a set of experiments evaluating a representational restriction on productions that guarantees an absence of expensive chunks, with encouraging results. We have applied our domain-independent abstraction mechanism to a set of problems in two domains (mobile robot and computer configuration), and evaluated its ability to reduce problem solving time, reduce learning time, and increase the generality of the rules learned. We have run a set of experiments which evaluate the ability of rules learned in medical diagnosis to transfer to related problems (done in a reduced-size version of NEOMYCIN-SOAR). In the area of theoretical developments and system building, we have extended our work on declarative learning to allow indexing off of arbitrary features, but in the process uncovered a new issue concerned with how to deal with multiple retrieval and discrimination.

Core System Development

- Because of budget cuts in our award, this has been a particularly busy and chaotic year in terms of changes to the orderly progression we had planned for the transition to a distributed environment. There were two immediate consequences of this cut: a) reducing our systems staff by two people and b) taking the DEC 2060 off of contract maintenance early in the grant year, thereby forcing us to close it down for routine use. These steps have had substantial impacts in forcing us to devote full energy to the 2060-to-SUN-4 transition mechanics approximately a year before we expected to be ready for it and in diverting staff from work on longer-term distributed computing problems. In spite of all this unplanned redirection of our energies, we have made substantial progress this past year as summarized below.
- Because of the necessary preoccupation of most of our staff with the premature 2060 transition this past year, we were not able to convene the visiting advisory group as was recommended by BRTP to help guide our long-term research efforts. As we finally close out the 2060 chapter this summer, we will plan to assemble such a group in the early fall (September or October) to reassess our plans for the coming two years.
- As detailed in our report last year, we have chosen Apple Macintosh II workstations as the general computing environment for researchers and staff, TI Explorer Lisp machines (including the microExplorer Macintosh coprocessor) as the near-term high-performance Lisp research environment, and a SUN-4 as the central network server replacement for the DEC 2060. We outlined there the many tasks facing us in making the transition from the central 2060 environment to the new distributed model, including selecting and integrating tools for text processing (editing, graphics, formatting, and bibliographic references), presentation graphics, printing, help facilities and distributed information access, interpersonal communication tools (EMail and BBoards), file management (storage, access, backup, and archiving), and system building tools (languages, development environments, and integration tools). Because of the high maintenance cost of the DEC 2060, we could not afford to continue its coverage in light of the large budget cut. Since this old-technology machine quickly becomes unreliable without regular maintenance, this forced us to transfer nearly all of our AIM community usage to the SUN-4/280 in October and November of 1988. The DECSystem-20 had been our major computer resource since February of 1983 and this machine, in turn, had replaced a KI-TENEX system in use for nine years earlier. Thus, our conversion to the UNIX based SUN-4/280 represented a major departure from a long-established approach to computing and for many, converting to the use of UNIX was a difficult transition. A significant and urgent effort went into developing a *UNIX Users Guide for TOPS-20 Users* which has provided substantial help in navigating through the most common of commands. In the process of

converting, we had to transfer about 400 user accounts. Most of the immediately-needed working files from the 2060 system were dumped to tape and loaded into the SUN-4. Most of this transfer was done during a four week period of intensive work. In addition, we had to orchestrate the transfer of "SUMEX-AIM" name from the 2060 to the SUN-4, and provide effective continuation of facilities such as EMail, BBoard. text processing, etc. Continuous and nearly compatible AIM community mail services were maintained through the transition by installing the Columbia University MM-C mail program on the SUN-4. This program closely duplicates the functions of the TOPS-20 COMAND JSYS under UNIX and presents the user with a mail reader/composer interface very similar to that of TOPS-20 MM. This system, coupled with a UNIX version of the EMACS text editor, called GNUEMACS, provided a relatively familiar setting for the most common computing functions used by AIM community members. In the succeeding months, we added bulletin board functionality to MM-C so that, from the user's perspective, mail access was nearly identical to the former system.

- Another major issue in the 2060-to-SUN-4 transition has been the need to provide our users with continued access to their large collection of archived files and to a set of permanent annual backup dumps (done January of each year) which have been collected and maintained since 1975. This has required very careful planning as the directory information for these tapes resides in Archive-Directory files (for TENEX) and the on-line File Descriptor Blocks (FDB's) of the TOPS-20 file system. These two sets of information must be converted to simple UNIX-compatible text files to provide users with continued facilities to review and access their collections of archived files. This work has been a major undertaking and is still in progress.
- In the move from TOPS-20 to UNIX, we have had to ensure continued access to "standard" services, such as file backup, archiving, a flexible and intuitive naming facility, and data interchange services (e.g., file transfer). UNIX has many of the needed facilities, e.g., backup, long names, hierarchical directory structure, some file property attributes, data conversion, and limited archival tools. We have worked on adapting a commercial system developed by UniTech to allow users to manage large file collections by moving files not needed on-line to and from off-line tape storage. This system also maintains a historical archive of files. The system is in beta test now and will be released to the entire community early this summer.
- Electronic mail continues as a primary means of communication for the widely spread SUMEX-AIM community. As reported last year, the move to workstations has forced a significant rethinking of the mechanisms employed to manage such mail in order to ensure reliable access, to make user addressing understandable and manageable, and to facilitate keeping the mail software distributed to workstations as simple, stable, and

maintainable as possible. We are following a strategy of having a shared mail server machine which handles mail transactions with mail clients running on individual user workstations. The mail server can be used from clients at arbitrary locations, allowing users to read mail across campus, town, or country. We have made significant progress this year in developing a Mac II version of the graphics-based MM-D/IMAP mail client reported on last year, including a complete rewrite of the InterLisp system into C and modifying the user reading and composing interface to be compatible with the Mac "look and feel". This system is nearly ready for alpha test starting early this summer.

One of the key issues in selecting the systems for our distributed computing environment was the performance of Common Lisp and to help make this evaluation, we have continued to expand an informal survey of the performance of two KSL AI software packages, SOAR and BB1, on a wide variety of machines. This study was completed this past year and a "final" report written (see Appendix B), recognizing that each month new workstations are announced that deserve additional evaluation.. Within a factor of two of the best performance, a considerable range of workstations based on stock microprocessor chips as well as specially microprogrammed Lisp chips have comparable performance. Even though performance gaps between microprogrammed Lisp systems and stock workstation implementations are narrowing, there still remains a significant difference in the quality of the development environments. We have attempted to distill the key features of the Lisp machine environments that would be needed in stock machine implementations in order to make them attractive in a development setting.

- This past year we acquired 2 NeXT workstations, primarily to understand and evaluate the power of the NeXT *Interface Builder* for AI software development. Integrating these prototype systems took a significant effort and they are now being used in several of our core research and applications projects. In addition, we have continued to support a limited number of other "standard" workstations for our work, including Mac II's, TI Explorers, and SUN's. We have continued to work toward a complete phase-out of our old Xerox and Symbolics Lisp machines.
- As reported last year, major changes in ARPANET service have been underway as ARPA has responded to its own budget pressures to reduce operating subsidy of the ARPANET. Starting late last spring, sections of the ARPANET serving university users were being shut down and replaced by connection to the NSFNET. Our own connection is just in the process of being decommissioned with our IMP scheduled to be removed sometime this summer. Our Internet access is now implemented through the Bay Area Regional Research Network (BARRNet) and the NSFNet. We continue to operate the Develcon gateway between our Ethernet environment and the TELENET network by which many AIM users gain access to SUMEX.

Other Resource Activities

- We have continued the dissemination of SUMEX-AIM technology through various media. The distribution system for our AI software tools (EMYCIN, AGE, and BB1) to academic, industrial, and federal research laboratories continues to work effectively. We have also continued to distribute the video tapes of some of our research projects including ONCOCIN, and an overview tape of Knowledge Systems Laboratory work to outside groups. Our group has continued to publish actively on the results of our research, including more than 45 research papers per year in the AI literature and a dozen books in the past 7 years on various aspects of SUMEX-AIM AI research. We assisted and participated actively in the AIM Workshop sponsored by AAAI and held at Stanford in 1988 and hosted a number of AIM community visitors at our Stanford research laboratory. Members of the Medical Computer Science group are participating in the early organization phases of another workshop during the spring of 1990.
- The Medical Information Sciences program, begun at Stanford in 1983 under Professor Shortliffe as Director, has continued its strong development over the past year. The specialized curriculum offered by the MIS program focuses on the development of a new generation of researchers able to support the development of improved computer-based solutions to biomedical needs. The feasibility of this program resulted in large part from the prior work and research computing environment provided by the SUMEX-AIM resource. As already reported, it has recently received enthusiastic endorsement from the Stanford Faculty Senate for an additional five years and has been awarded renewed post-doctoral training support from the National Library of Medicine with high praise for the training and contributions of the SUMEX-AIM environment from the reviewing study section. This past year, MIS students have published many papers, including several that have won conference awards.
- We have continued to recruit new user projects and collaborators to explore further biomedical areas for applying AI. A number of these projects are built around the communications network facilities we have assembled, bringing together medical and computer science collaborators from remote institutions and making their research programs available to still other remote users. At the same time we have encouraged older mature projects to build their own computing environments thereby facilitating the transition to a distributed AIM community. A substantial number of projects have already moved to their own computing resources.
- SUMEX user projects have made good progress in developing and disseminating effective consultative computer programs for biomedical research. These systems provide expertise in areas like cancer chemotherapy protocol management, clinical diagnosis and decision-

making, and molecular biology. We have worked hard to meet their needs and are grateful for their expressed appreciation (see Section IV).

III.A.2.3. Core ONCOCIN Research

ONCOCIN is a data management and therapy advising program for complex cancer chemotherapy experiments. The development of the system began in 1979, following the successful generalization of MYCIN into the EMYCIN expert system shell. The ONCOCIN project has evolved over the last eight years: the original version of ONCOCIN ran on the time-shared DECSys-20 computers using a standard terminal for the time-oriented display of patient data. The current version uses compact workstations running on the Ethernet network with a large bit-mapped displays for presentation of patient data. The project has also expanded in scope. There are three major research components: 1) ONCOCIN, the therapy planning program and its graphical interface; 2) OPAL, a graphical knowledge entry system for ONCOCIN; and 3) ONYX, a strategic planning program designed to give advice in complex therapy situations. Each of these research components has been split into two parts: continued development of the cancer therapy versions of the system, and generalization of each of the components for use in other areas of medicine. This portion of the annual report will concentrate on the *generalization tasks* related to treatment planning, knowledge acquisition, and research to extend ONCOCIN for application in clinical trial domains other than medical oncology (E-ONCOCIN).

In addition, we will discuss the continued development of a generalized knowledge acquisition tool designed to encode descriptions of clinical trials. The system, named PROTÉGÉ, was the Ph.D. thesis work of Mark Musen, (who joined our faculty this year). The output of PROTÉGÉ is an OPAL-like input system designed for a target clinical area such as hypertension. This input system (HTN-OPAL) can then be used to create the hypertension knowledge base for an E-ONCOCIN like system. This experiment was carried out this year for both the hypertension and oncology domains. Details of this project are described later in this report.

(1) Overview of the ONCOCIN Therapy Planning System

ONCOCIN is an advanced expert system for clinical oncology. It is designed for use after a diagnosis has been reached, focusing instead on assisting with the management of cancer patients who are receiving chemotherapy. Because anticancer agents tend to be highly toxic, and because their tumor-killing effects are routinely accompanied by damage to normal cells, the rules for monitoring and adjusting treatment in response to a given patient's course over time tend to be complex and difficult to memorize. ONCOCIN integrates a temporal record of a patient's ongoing treatment with an underlying knowledge base of treatment protocols and rules for adjusting dosage, delaying treatment, aborting cycles, ordering special tests, and similar management details. The program uses such knowledge to help physicians with decisions regarding the management of specific patients.

A major lesson of past work in clinical computing has been the need to develop methods for integrating a system smoothly into the patient-care

environment for which it is intended. In the case of ONCOCIN, the goal has been to provide expert consultative advice as a by-product of the patient data management process, thereby avoiding the need for physicians to go out of their way to obtain advice. It is intended that oncologists use ONCOCIN routinely for recording and reviewing patient data on the computer's screen, regardless of whether they feel they need decision-making assistance. This process replaces the conventional recording of data on a paper flowsheet and thus seeks to avoid being perceived as an additive task. In accordance with its knowledge of the patient's chemotherapy protocol, ONCOCIN then provides assistance by suggesting appropriate therapy at the time that the day's treatment is to be recorded on the flowsheet. Physicians maintain control of the decision, however, and can override the computer's recommendation if they wish. ONCOCIN also indicates the appropriate interval until the patient's next treatment and reminds the physician of radiologic and laboratory studies required by the treatment protocol.

(2) Implementation of the ONCOCIN Workstation in the Stanford Clinic

In mid-1986, we placed the workstation version of ONCOCIN into the Oncology Day Care clinic. This version is a completely different program from the version of ONCOCIN that was available in the clinic from 1981-1985 — using protocols entered through the OPAL program, with a new graphical data entry interface, and revised knowledge representation and reasoning component. One person in the clinic (Andy Zelenetz) became primarily responsible for making sure that our design goals for this version of ONCOCIN were met. His suggestions included the addition of key protocols and the ability to have the program be useful for clinicians as a data management tool if the complete treatment protocol had not yet been entered into the system. Additional fellows were trained on a very stable release of ONCOCIN that became available in early 1988. A version of the system was sent to the University of Pittsburgh for evaluation and to the National Library of Medicine Artificial Intelligence Demonstration Center. For these various efforts, Janice Rohn has created an extensive user manual, sample patient interactions, and reminder cards to shorten the training period for ONCOCIN.

The process of entering a large number of treatment protocols in a short period of time led to other research topics including: design of an automated system for producing meaningful test cases for each knowledge, modification of the design of the time-oriented data base and the methods for accessing the data base, and the development of methods for graphically viewing multiple protocols that are combined into one large knowledge base. These research efforts will continue into the next year. In addition, some of the treatment regimens developed for the original mainframe version are still in use and can be transferred to the new version of ONCOCIN.

We also received new insights about the design of the internal structures of the knowledge base (e.g., the relationship between the way we refer to

chemotherapies, drugs, and treatment visits). We will continue to optimize the question-asking procedure, the method for traversing the plan structure in the knowledge base, and consider alternative arrangements used to represent the structure of chemotherapy plans. Although we have concentrated our review of the ONCOCIN design primarily on the data provided by additional protocols, we know that non-cancer therapy planning problems raise similar issues. The E-ONCOCIN effort is designed to produce a domain-independent therapy planning system that includes the lessons learned from our oncology research.

(3) E-ONCOCIN: Domain Independent Therapy Planning

During the past two years, our E-ONCOCIN research has concentrated on understanding how protocols in medicine vary across subspecialties. We are examining several application areas: the intensive care unit, insulin treatment for diabetes, hypertension protocols, and both standard and complex cancer treatment problems. The diagnosis and therapy selection for patients in the intensive care unit is a natural application area because it is based on changing data and the need to determine the response to therapy interventions. In addition, it is an area where reasonable mathematical models of the respiratory system can be integrated into the expert system (see description of the VentPlan system). We also felt that the area of insulin treatment for diabetes would be a good area to explore. Like cancer chemotherapy, the treatments for diabetes continues over a long period of time and has been the area of intensive protocol development. Unlike cancer chemotherapy, the treatment plan must handle multiple treatments in one day and deemphasises the use of multiple drugs (although there are a variety of types of insulin). During 1987, using the medical literature and several internists in the medical computer science research group (Mark Frisse, Mark Musen, and Michael Kahn), we performed knowledge acquisition experiments for insulin treatment of diabetes. The proposed structure for the knowledge base was implemented using the object-oriented programming language upon which ONCOCIN has been based. These experiments, like those of adding more protocols to ONCOCIN, demonstrated the need for changes in the way that the knowledge base can access the time-oriented data base that records patient data and previous conclusions. The relationships between the different doses and types of insulin treatments will also require alternative ways of building treatment hierarchies. Thus, our initial experiments have shown that many of the elements of the ONCOCIN design are sufficiently general for other application areas, but that some specific elements (particularly the representation of temporal events) will have to be generalized. A description of our revised temporal representations has appeared in a thesis by Michael Kahn, who is at Washington University in St. Louis, based on work completed at Stanford and U.C.S.F.

A logical extension of Kahn's work has been an investigation of how to modify the EONCOCIN framework so that it can work with established data base tools instead of the hand-tailored data base currently in use. In making this

transformation, we need to maintain the access to data that is mediated by the temporal network, although most relational data bases are not organized for encoding temporal information. In this work, we must be able to describe the changing clinical context and event intervals that show up in many diverse application areas. An example of a new area that we are exploring is the treatment of AIDS patients on clinical protocols. While this area is similar to some aspects of oncology protocols, we are faced with significant differences in the way that treatment is delivered. AIDS patients do not always follow the type of strict temporal schedules (e.g., regular visits to outpatient clinics) seen with oncology patients. They have a chronic disease with acute exacerbations of opportunistic infections. Medications are often given orally as opposed to the controlled intravenous infusions of medical oncology patients. Furthermore, the medication schedule is interrupted by frequent hospitalizations and confounded by taking drugs not on the protocol. Together, these factors will require a much more flexible model of the temporal dimension of treatment planning.

(4) OPAL: Graphical Knowledge Acquisition Interface

OPAL is a graphical environment for use by an oncologist who wishes to enter a new chemotherapy protocol for use by ONCOCIN or to edit an existing protocol. Although the system is designed for use by oncologists who have been trained in its use, it does not require an understanding of the internal representations or reasoning strategies used by ONCOCIN. The system may be used in two interactive modes, depending on the type of knowledge to be entered. The first permits the entry of a graphical description of the overall flow of the therapy process. The oncologist manipulates boxes on the screen that stand for various steps in the protocol. The resulting diagram is then translated by OPAL into computer code for use by ONCOCIN. Thus, by drawing a flow chart that describes the protocol schematically, the physician is effectively programming the computer to carry out the procedure appropriately when ONCOCIN is later used to guide the management of a patient enrolled in that protocol.

OPAL's second interactive mode permits the oncologist to describe the details of the individual events specified in the graphical description. For example, the rules for administering a given chemotherapy will vary greatly depending upon the patient's response to earlier doses, intercurrent illnesses and toxicities, hematologic status, etc. For example, one form permits the entry of an attenuation schedule for an agent based upon the patient's white count and platelet count at the time of treatment. Tables such as this are generally found in the written version of chemotherapy protocols. Thus OPAL permits oncologists to enter information using familiar forms displayed on the computer's screen. The contents of such forms are subsequently translated into rules and other knowledge structures for use by ONCOCIN.

Status of the OPAL System

The OPAL is one of the few graphical knowledge acquisition systems ever designed for expert systems. Even fewer are designed to be used as the main method for entering knowledge as opposed to a proof of concept implementation. We have pursued three directions in the development of the OPAL system, also in response to the large number of protocols entered through this system during the several years.. The first direction is the modification of graphical forms needed to allow the entry of facts that did not show up in the protocols used to test the initial version of OPAL. OPAL continues to assume that most of the knowledge to be entered will have very stereotyped forms, e.g., dose attenuations for most treatment toxicities are based on a comparison of only one laboratory measurement at a time, such as using the BUN to adjust for renal toxicity. We sometimes need much more complex ways of stating the scenarios in which dose adjustments may be necessary. This need has led us in a second direction, towards a "lower-level" rule entry approaching the syntax of the reasoning component of ONCOCIN, but using graphical input devices where applicable. A major accomplishment of this last year was to experimentally combine the OPAL and ONCOCIN programs into one working program, and to completely enter knowledge from OPAL using both the high level tools and lower level rule editors, but without needing to make changes at the ONCOCIN side of the system. The OPAL program maps the information provided on the graphical forms into a complex data structure (called the IDS) that represents the required knowledge to specify the contents of a protocol. This data structure is used for copying information from one protocol to another, and as the basis for the creation of the ONCOCIN knowledge base. Our experiments with OPAL, and our intention to generalize OPAL use outside of oncology protocols, suggests that we reorganize the OPAL program to use a relational data base to store its knowledge. We have patterned the data base after an existing data base query syntax. Because no data bases exist for the InterLisp language upon which OPAL is based, we reimplemented the data base from its written description. We continue to explore the appropriate avenue for the connection of our knowledge acquisition systems to data bases, and have concentrated on the SQL query language to a relational data base using the client-server model (e.g., the physical data base may exist on a different machine than the knowledge acquisition tool — transmitting the query and the response over the network).

With the future of dedicated lisp processors looking very unclear, we began to explore alternative platforms for developing the interface for OPAL-like systems. We have begun experiments using HyperCard on the Mac II and Interface Builder on the NeXT machine. In order to build experience with the each of these possible platforms, we have re-implemented portions of OPAL system, and are analyzing the results. It is particularly hard to determine the best platform since the NeXT machine software is still in a rudimentary

stage, and HyperCard on the Mac II has significant limitations including small "card size" and the inability to display multiple cards simultaneously.

(5) Generalized Knowledge Acquisition through PROTÉGÉ

Mark Musen designed and implemented the first version of the PROTÉGÉ knowledge-acquisition-system development tool. PROTÉGÉ is used to collect information which describes the concepts (both entities and their relationships) in an application area for which a skeletal-planning type of expert system would be useful, concentrating on clinical trials. The system acquires the "ontology" of a domain through a series of fill-in-the-blank forms and a "flowchart-entry" tool. These concepts are then mapped onto a set of generic forms, which, in turn, create a knowledge acquisition tool for the application area.

PROTÉGÉ makes use of the forms management system built for the original OPAL, and a newly developed relational data base management system written for the Xerox InterLisp-D workstations. The output of PROTÉGÉ is an OPAL-like set of forms tailored to the special structures of the application area. To test these ideas, we first reimplemented portions of the OPAL interface from a high level description of oncology. After the translation process to the ONCOCIN reasoning program, a consultation was run that matched the manually built system. This experiment was then repeated for the area of hypertension protocols for which ONCOCIN had never been specifically designed. With some minor generalizations to the ONCOCIN reasoner and interviewer, we were able to run a hypertension consultation. With Mark returning as a faculty member, this work has continued this year, but faces the same platform and basic tool issues as described above. Portions of the PROTÉGÉ interface is being reimplemented using the Interface Builder on the NeXT computer.

(6) Speech Input to Expert Systems

(6.1) Prototype Speech Hardware/Software System

In 1987 we began a project to explore the integration of speech-recognition technology into the interface to ONCOCIN running on the XEROX Lisp workstations. The project uses a commercially available continuous-speech-recognition product loaned by the vendor, Speech Systems, Inc. (SSI) of Tarzana, California. The speech recognizer consists of a custom processor, called the Phonetic Engine® and a suite of software modules called the Phonetic Decoder™. The Phonetic Decoder™ initially ran on a SUN 3/75 and now runs on the NeXT computer.

The development of this project requires significant experience in distributed computing since the phonetic device, initial parsing software, and the ONCOCIN system all reside on different pieces of hardware. One of the early steps is to allow the Lisp machine to remotely control the parsing software on the SUN. We built an interpreter for communicating between the speech software library running on the SUN and the Xerox Lisp machine. This

interpreter reads Lisp-style function calls corresponding to the speech library routines and returns Lisp-style results as remote procedure call (RPC) mechanism. We then wrote the library of corresponding Lisp stub routines and a function to connect to the SUN workstation and start the server. In normal operation, we call the C-based speech library routines from Lisp as if they were Lisp functions. During this year, we were able to port a version of the SSI system to the NeXT machine. In addition, we mounted a new version of the speech hardware (PE200) such that we can compare the two versions of the hardware with each other in terms of accuracy, speed, and ease of use. In addition, we were able to obtain a copy of the CMU speech understanding system (running on the NeXT machine), and were able to make some comparisons with the SSI speech processing hardware. All of these systems require extremely fast processors in order to deliver reasonable response time. The announcement of relatively inexpensive and fast RISC-based architectures should enhance the acceptance of speech input systems. Because some of the applications of the speech equipment do not require continuous speech, we are also evaluating this mode.

We created a prototype system that permits users to navigate the graphical interface and enter clinical data using speech. The system uses the location of the cursor on the screen to provide a context for choosing candidate grammars with which to attempt recognition of a user's utterance. The system dynamically re-orders the list of candidate recognition grammars based on the dialog history. Albeit with limitations on the legal grammars, it is now possible to carry on most of the ONCOCIN data acquisition steps using speech alone or speech plus pointing with the mouse. In addition, some elements such as the neural toxicities can be entered as textual descriptions and automatically encoded as one the 1-4 point scale used on flowsheet forms. This system was extended such that it was possible to have an entire ONCOCIN consultation only with voice commands.

In order to translate an utterance back into an action that can occur in the ONCOCIN interface, we need the ability to reparse the text string returned by the SSI equipment. The SSI equipment uses (potentially complex) syntax, built up of various classifications, to understand sentences but returns just the ASCII component of the actual sentence; you can not get it in terms of the original classifications in the syntax (which are generally semantically significant). When the ASCII string is returned, a description of the syntax is used to convert the string into a parse tree that relates directly to the grammar definition. We can now process the returned information at a much higher level than was possible with the simple ASCII text. In addition, we have built tools to perform the semantic analysis that is converted to specific actions in the interface language. For example, the utterance "white blood count is 3.2" is parsed using a grammar "<parameter> is <value>." When the string is recognized, it must be turned back into the action sequence — in this case, opening the hematology form if it is not already open, and highlighting the WBC row, and placing the value 3.2 in the column corresponding to the current visit. While the graphical effects appear simple,

these internal transformations may be quite complex. We are deriving from this experience a description of an interface manipulation language that details the "legal operations" in an ONCOCIN-like graphical interface.

We have also explored a second medical record-keeping task — the creation of portions of a progress note that describes in textual form the changes in the patients status from week to week. We have developed a system that uses broad categories of data as specified on the spreadsheet as a prompt for textual input. For example, the spreadsheet includes a line for "gastrointestinal toxicity" of grades 1 to 4. This becomes the topic of a sentence to be included in the progress report such as "The patient has experienced nausea and vomiting one to three times per day over the last week." The number of sentences that are possible cannot practically be displayed on the screen, so we are experimenting with various types of prompting to give the user a sense of what sentences the system will be able to recognize; e.g., by selecting random examples from the grammars to present as hints for entry or by presenting a graphical version of the legal syntax at any point. This system was further extended to develop a portion of the physical examination (PE) portion of the progress note. The PE consists of a short description about each organ system in the body, concentrating on those specific aspects that are remarkable during a patient visit (changes or significant findings). Using hypertext techniques similar to those found in outlining programs such as MORE, we created a hierarchical description of the breast portion of the PE, and have been able to create reasonable interactions for this one segment of the PE. As the user becomes more familiar with the structure of the legal syntax and the descriptions necessary to reach a particular level of detail, then they can use speech to bypass the graphical interface and directly enter the utterance. We are exploring ways to scale these techniques up to allow for entry of the entire PE, while examining how to organize the less structured elements of the progress note.

(6.2) Speech Experiments

We are performing experiments to (1) enhance the system's grammars with a wider range of phrases clinicians actually use when talking to a computer and (2) gain insights into clinicians' models of spoken interaction with advice systems so that we may ground our interface design in observed practice. In order to assess how physicians would speak to a computer in an ideal situation without constraints or prior assumptions, we are conducting a series of experiments which simulate continuous-speech understanding by computers. The setting of these experiments includes a hidden computer operator simulating the output of ONCOCIN if it had the ability to understand the spoken input, as well as a video camera to record both audio and visual clues. Typed responses from the operator are translated back as actions on the computer display as well as audio responses through the use of a speech synthesizer. It appears to the subject as if the computer is understanding and responding to their speech. The physicians use

ONCOCIN in the same manner as it is used in the clinic when they see patients, but with the added capability of speech input.

These experiments enable us to both build up a basic vocabulary for the speech system as well as examine subtle linguistic issues to guide future directions. The experiments have been completed and we are analyzing the data in order to describe user-specific grammars, and to see how individuals react to purposeful misunderstandings by the computer. We are experimenting with a range of possible misunderstandings (e.g., from "What did you say?" to "The platelet count was WHAT?"). Our initial impressions of these experiments were that each subject quickly developed her own subgrammar for entering the information, often based on their belief about the system's knowledge of the domain, and that subjects responded directly to misunderstandings with partial phrases in a manner similar to being asked by a human. This experiment underscores the importance of being able to obtain not only the top-rated sentence from the recognition system, but also an indication as to which parts of the utterance were most likely to have been misrecognized.

(7) Object Language Support for ONCOCIN Project

We released a new version of our object language at the start of this past year which has proven to be the most stable and powerful version to date. There have been a number of minor bug fixes and several feature additions over the course of the year but for the most part the system as required much less attention than in previous years. The number of new systems being built on it (like our speech work) continues to increase. Future planning for the system consists of determining whether or not it should be converted to Common Lisp, based on whether object systems available under Common Lisp are sufficient for our needs, and if we do convert it what it would look like if properly integrated with that language.

(8) Personnel

Samson Tu has been primarily responsible for the design of E-ONCOCIN, Michael Kahn developed the temporal representations used by the system, Clifford Wulfman has been involved with extensions to the data entry interface and the extensions to the interface in order to add speech input. Samson and Cliff were responsible for extensions to their programs to support the PROTÉGÉ effort. David Combs has been involved with the knowledge acquisition interface and provided major programming support for the PROTÉGÉ effort. Janice Rohn has been involved with the entry of protocols, interaction with physicians using the system, documentation of the system, and execution of the speech experiments. Christopher Lane has developed the object-oriented systems software upon which the entire ONCOCIN system is designed. He has been instrumental in developing the systems software for the speech project, and has performed our evaluations of different speech input devices and configurations. Ellen Isaacs, a Ph.D. student in Psycholinguistics has helped to design the simulated speech-input

experiments. Monica Rua has developed the progress note software in conjunction with Cliff and Christopher.

III.A.2.2. Core AI Research

(1) Rationale

Artificial Intelligence (AI) methods are particularly appropriate for aiding in the management and application of knowledge because they apply to information represented symbolically, as well as numerically, and to reasoning with judgmental rules as well as logical ones. They have been focused on medical and biological problems for well over a decade with considerable success. This is because, of all the computing methods known, AI methods are the only ones that deal explicitly with symbolic information and problem solving and with knowledge that is heuristic (experiential) as well as factual.

Expert systems are one important class of applications of AI to complex problems — in medicine, science, engineering, and elsewhere. An expert system is one whose performance level rivals that of an human expert because it has extensive domain knowledge (usually derived from an human expert); it can reason about its knowledge to solve difficult problems in the domain; it can explain its line of reasoning much as an human expert can; and it is flexible enough to incorporate new knowledge without reprogramming. Expert Systems draw on the current stock of ideas in AI, for example, about representing and using knowledge. They are adequate for capturing problem-solving expertise for many bounded problem areas. Numerous high-performance, expert systems have resulted from this work in such diverse fields as analytical chemistry, medical diagnosis, cancer chemotherapy management, VLSI design, machine fault diagnosis, and molecular biology. Some of these programs rival human experts in solving problems in particular domains and some are being adapted for commercial use. Other projects have developed generalized software tools for representing and utilizing knowledge (e.g., EMYCIN, UNITS, AGE, MRS, BB1, and GLisp) as well as comprehensive publications such as the three-volume Handbook of Artificial Intelligence¹ and books summarizing lessons learned in the DENDRAL and MYCIN research projects².

This report documents progress on the basic or core research activities within the Knowledge Systems Laboratory (KSL), funded in part under the SUMEX resource as well as by other federal and industrial sources. This work explores a broad range of basic research ideas in many application settings,

¹ Barr, A., Cohen, P. R., and Feigenbaum, E. A. *The Handbook of Artificial Intelligence, Volumes I, II, and III*. William Kaufmann, Inc., Los Altos, CA, 1981 and 1982.

² Buchanan, B. G., and Shortliffe, E. H. *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA, 1984.

Lindsay, R. K., Buchanan, B. G., Feigenbaum, E. A., and Lederberg, J. *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project*. McGraw-Hill, New York, NY, 1980.

all of which contribute in the long term to improved knowledge based systems in biomedicine.

(2) Highlights of Progress

In the last year, research has progressed on several fundamental issues of AI. As in the past, our research methodology is experimental; we believe it is most fruitful at this stage of AI research to raise questions, examine issues, and test hypotheses in the context of specific problems, such as management of patients with Hodgkin's disease. Thus, within the KSL we build systems that implement our ideas for answering (or shedding some light on) fundamental questions; we experiment with those systems to determine the strengths and limits of the ideas; we redesign and test more; we attempt to generalize the ideas from the domain of implementation to other domains; and we publish details of the experiments. Many of these specific problem domains are medical or biological. In this way we believe the KSL has made substantial contributions to core research problems of interest not just to the AIM community but to AI in general.

Progress is reported below under each of the major topics of our work. Citations are to KSL technical reports listed in the publications section.

(2.1) Large Multi-use Knowledge Bases for Science and Engineering

There is considerable power in the current stock of AI techniques, as exemplified by the rate of transfer of ideas from the research laboratory to commercial practice. But we also believe that today's technology needs to be augmented to deal with the complexity of medical information processing. One of our core research goals is to analyze the limitations of current techniques and to investigate the nature of methods for overcoming them. Long-term success of computer-based aids in medicine and biology depend on improving the programming methods available for representing and using domain knowledge. That knowledge is inherently complex: it contains mixtures of symbolic and numeric facts and relations, many of them uncertain; it contains knowledge at different levels of abstraction and in seemingly inconsistent frameworks; and it links examples and exception clauses with rules of thumb as well as with theoretical principles. Moreover, strategies for using domain knowledge can be complex as well, particularly in dynamic environments which require continual reassessment of the best use of limited computational resources. Current techniques have been successful only insofar as they severely limit this complexity. As the applications become more far-reaching, computer programs will have to deal more effectively with richer expressions, more voluminous amounts of knowledge and more complex control strategies.

Expert systems are being developed that impact nearly every field of human endeavor: medicine, manufacturing, financial services, diagnosis of machinery, geology, molecular biology and structural design, to name a few. Each new instance is a confirmation of the Knowledge Principle (knowledge is power). In each system, expert level problem-solving performance is obtained

by using relatively simple and uniform reasoning methods which access an extensive body of domain knowledge. The ability of these systems lies primarily in the *specific* concepts, facts, methods, models, etc. that can be brought to bear on the problem. A corollary to the Knowledge Principle is that significant improvements in the power of knowledge-based systems will be derived primarily from the ability to access large amounts of knowledge.

To test that corollary, we are embarked on a multi-year research effort that will develop methods for building *large, multi-use knowledge bases* (LMKB). We believe construction of a LMKB is an essential step toward resolving two fundamental problems plaguing the current generation of expert systems. The first is *brittleness*: current systems can exhibit only a very narrow range of expert behavior, and their performance falls off precipitously at the limits of their expertise. The second problem is *over-specialization*: a knowledge base constructed to support of one type of expert task (e.g., diagnosis) cannot be used to support other types of tasks (e.g., design).

Our hypothesis is that both the problems of brittleness and over-specialization can be addressed by constructing large, multi-use knowledge bases. A LMKB would:

- 1) encode domain knowledge in greater depth and breadth than required for any specific task,
- 2) encode knowledge that cuts across many domains of expertise, and
- 3) serve as a core repository of knowledge to be accessed by large numbers of specific applications.

Research of this scope raises many important research issues. Of primary importance are issues of *knowledge representation*. The AI field needs to broaden its understanding of how to give programs a representation of the physical world, its phenomena, its processes, and its devices (in addition, of course, to the conventional mathematical/numeric representations of scientific computing). These issues are both epistemological and technological (how the concepts are named, described, and related; and how they are stored for efficient access and use by reasoning processes). Of these, the epistemological issues are key, because we intend our knowledge bases to be long-lasting, robust, and built upon by many other groups. The "large" in large knowledge bases cannot become a reality unless there is cumulation, and cumulation will only come about if we demonstrate the epistemological adequacy of our pioneering efforts at representing the world of physics and engineering. During the past year we have been exploring a variety of representations and the systems which employ them, including CYC from MCC, CLASS from Schlumberger, and QPE from Univ. of Illinois.

A second major issue is one we call *knowledge compilation*. This is the bridge between the second-era systems and the expert systems of the first era. Effective problem solving is not typically carried out at the level of first principles, but rather at the level of more compact, efficient forms of knowledge, compiled from experience with specific tasks. So-called

knowledge compilers are needed to translate from the more general forms of knowledge (e.g. the basic principles of the physical science and engineering) to highly specific forms needed for diagnosis, design, etc. During the initial year of this project, we demonstrated the feasibility of this approach by deriving a set of diagnostic rules and a set of redesign heuristics from a single knowledge base containing a model of the structure and behavior of the Reaction Wheel Assembly mentioned above. We are also developing an integrated scheme for using "first principles" knowledge of the physical world for simulation. Given a description of the structure of a device in terms of its constituent objects and their relations, the system identifies applicable physical laws, processes, types of matter, etc. and produces a set of equations to describe the behavior of the device. The equation model is analyzed using the method of causal ordering to produce a model that reveals the dependency relations among the parameters of the model¹.

Closely related to knowledge compilation is its inverse, *knowledge justification*. That is, how can one justify the highly specialized knowledge in a typical task-specific expert system in terms of more fundamental laws of nature and/or principles of engineering? Our approach here is to treat both the compilation and justification processes together. As knowledge is compiled into a more specialized form, the system will record the links back to the more fundamental sources of knowledge. A side benefit of knowledge justification will be the ability to provide more satisfactory explanations of an expert system's line of reasoning than just a trace of the rules that were fired.

A fourth research issue is concerned with *model-based reasoning*, and in particular, reasoning about physical devices using multiple approximate models. Scientists and engineers have the ability to model the physical world at different levels of detail. For example, we can solve problems of motion in a gravitational field by assuming objects of no size (point masses) moving in a vacuum. This is an appropriate model for predicting the fall-time of a rock but not of a feather. Having a detailed model of the domain under consideration allows one to reason correctly in a wide range of situations, including previously unanticipated ones. However, this robustness may entail an excessive computational cost. A diagnostic program that used a detailed model of the human metabolic system, for example, would be a very powerful tool but may be prohibitively costly to run. How does one select the appropriate model? What solution method should be used with each model? How can a reasoning system recognize that it must use an alternative model, and how does one shift from one model to another? These issues are currently being investigated in the context of the electrical power system on the Hubble Space Telescope (HST). During the past year we have begun to investigate methods of controlling the expense of model-based reasoning by generating multiple, approximate models of the task domain, and developing methods for choosing, from this space of possible models, the simplest model

¹ Iwasaki, Y., and Simon, H. A. "Causality in Device Behavior." *AI Journal*, 29:3-32, 1986.

that provides acceptable answers. Our proposed approach to tackling this problem is to explicitly represent the assumptions underlying each model, and to use these explicitly represented assumptions to help pick the appropriate model.

Finally, we must learn to integrate different types of reasoning in physical domains. For example, *case-based reasoning* is widely used in several engineering domains, especially in the design of new artifacts. Designers search for a previous design that most closely matches the current requirements, and then make appropriate modifications. However, even when past cases are used to construct a new solution, model-based reasoning is needed to make the modifications to fit the current problem, and also to detect unforeseen effects of those modifications..

(2.2) Adaptive Intelligent Systems

How can we design flexible control structures for powerful problem solving programs? How can we use these structures effectively in many problem domains? How can we represent processes and reason about their behavior, and perform intelligent actions under *real-time* requirements?

We have continued to develop the BB1 blackboard architecture to address these and related problems. In particular, we have begun or continued work on the following domain-independent BB1 modules:

- The Focus module provides a dynamic focus of attention and protects the application system from input data overload. It continuously monitors all sensors, abstracts sensed data (e.g., as value categories, averages, trends, or patterns), and filters the abstracted data before relaying it to the BB1 application system.
- The ReAct module provides time-sensitive problem detection and response capabilities. It propagates asynchronously sensed data through a hierarchically partitioned belief network. It uses time and other resource constraints to determine whether to continue the analysis or act on the basis of its current assessment.
- The ICE module provides reasoning from first principles to handle complex or unfamiliar problems. It uses structure/function representations of generic physical systems (e.g., flow, diffusion) and particular domain systems (e.g., respiration, circulation) to diagnose, explain, and predict problems.
- The TPlan module provides time-sensitive planning of coherent courses of action. It plans actions forward in time and at successively more detailed levels of abstraction. It uses time and other resource constraints to determine whether to continue plan refinement or act on the basis of its current plan.
- The TDB module provides a temporally organized database of observed, expected, and intended models of external entities, and associated temporal reasoning functions.

In collaboration with Dr. Adam Seiver of the Palo Alto VAMC, we have applied these capabilities in the Guardian system for intensive care monitoring. Our work on Guardian is reported in section IV of this progress report.

We have given demonstrations of the Guardian system to many colleagues in the medical AI and larger AI communities, for example to: Dr. Lawrence Fagan and his students from Stanford's Medical Computing Systems Group; Dr. Seppo Kalli, Director of Medical Signal Processing at Technical Research Centre of Finland; Dr. William Pardee and his associates from the Rockwell Science Center; Dr. Perry Thorndyke and his associates from FMC Corporation; Dr. Joseph Naser of the Electric Power Research Institute.

We have made (or plan to make) invited presentations of this research to: IEE Workshop on Expert Control Systems, Brighton England, June, 1990; International Joint Conference on Artificial Intelligence, Detroit, Aug, 1989; AI Systems in Government Conference, Washington D.C., March, 1989; AAI Symposium on Knowledge System Development Tools, Stanford, March, 1989; Stanford SIGLunch, February, 1989; Workshop on Formal Aspects of Semantic Networks, Catalina, February, 1989; Carnegie Symposium on Architectures for Intelligence, Pittsburgh, May, 1988; Advanced Decision Systems, Palo Alto, May, 1988; Boeing Computer Services, Bellevue, WA., March, 1988; DARPA Knowledge-Based Planning Workshop, Austin, December, 1987.

The project has produced the following recent publications, including KSL 88-64, KSL 89-05, KSL 89-06, and:

Hayes-Roth, B., Hewett, M., Washington, R., Hewett, R., and Seiver, A. **Distributing intelligence within a single individual.** In L. Gasser and M.N. Huhns (Eds.) *Distributed Artificial Intelligence* Volume 2. Morgan Kaufmann, 1989.

Hewett, R., and Hayes-Roth, R. **Representing and reasoning about physical systems using generic models.** In J. Sowa (Ed.) *Formal Aspects of Semantic Networks*. Morgan Kaufmann, 1989.

Hayes-Roth, B. **Dynamic control planning in adaptive intelligent systems.** *Proceedings of the DARPA Knowledge-Based Planning Workshop*, 1989.

(2.3) Advanced Architectures

The goals and technical approach of this project, largely supported by DARPA under the Strategic Computing Program, have been discussed in previous annual reports. In the 1988 Annual Report we described the various components of the project in some detail, and reported on the current state of progress. The discussion here will be limited to reporting progress in the past year on each of the components. An overview of the entire project has recently been written (KSL 88-71), and contains a comprehensive bibliography of publications produced by the project.

SIMPLE/CARE Multiprocessor Simulation System

SIMPLE/CARE is a powerful simulation system which permits empirical studies of expert system performance on a wide class of multicomputer architectures, including quantitative measurements of system behavior. It forms the foundation for our empirical investigations of software architectures and hardware system architectures for concurrent knowledge-based systems. SIMPLE is a CAD (Computer Aided Design) system for hierarchical, multiple level specification of computer architectures and includes an associated mixed-mode, event-based simulator. CARE is a parameterized, multiprocessor array emulation specified in SIMPLE's specification languages and running on SIMPLE's simulator. Our simulation system is in use by several research groups at Stanford, and it has been ported to several external sites including NASA Ames Research Center. A videotaped tutorial was held in June, 1988, attended by representatives from industry and government, which described the CARE/SIMPLE system, as well as the LAMINA programming interface (see below). The attendees received instruction in use of the system for making measurements of the performance of various simulated multiprocessor applications.

Due to rapidly growing interest in the SIMPLE/CARE system, a major effort is now underway to port it to wider class of hardware platforms. The system is currently being reimplemented in Common Lisp and the X window system, with the Sun workstation as the initial target.

During the past year, the research effort associated with SIMPLE/CARE has largely focussed on investigations of communication protocols (KSL 88-81) and techniques for monitoring concurrent object-based applications (KSL 89-15).

LAMINA Programming Interface

LAMINA provides extensions to Lisp for studying expressed concurrency in functional programming, object oriented, and shared variable models of concurrent computation. The implementation of the support for all three computational models is based on the common notion of a stream, a data type which can be used to express pipelined operations by representing the promise of a (potentially infinite) sequence of values. LAMINA also provides system support for the management of software pipelines and dynamic structure creation, relocation, and reclamation in a multiprocessor, multi-address-space system. Algorithms and applications written in LAMINA may be run on the SIMPLE/CARE simulation system in order to study their execution on alternative multiprocessor architectures.

The development of LAMINA was completed over a year ago and is being ported to Common Lisp along with the SIMPLE/CARE system. During the past year, the shared variable model was used as the basis of a shared memory Lisp package, called QL (KSL 88-85).

CAGE and Poligon Problem Solving Frameworks

CAGE, a framework for building and executing applications as a concurrent blackboard system, was described in last year's annual report. Its development was essentially completed during the past year. The Poligon problem solving framework, for the development of Blackboard-like applications on a (simulated) multiprocessor, was also described in the last annual report. Its development is now essentially complete and documentation for the system is now available (KSL 88-69 and KSL 89-37). Reference manuals for both CAGE and Poligon are being updated, and the software itself is undergoing some minor changes to bring it in line with the documentation, in preparation for external delivery.

CAGE, Poligon and LAMINA Comparative Experiments

As described in last year's report, a series of end-to-end experiments comparing various concurrent programming systems for knowledge-based applications was conducted. The goals of these experiments are to:

- Obtain quantitative comparisons of the performance of the programming systems.
- Gain insights into how different concurrent programming models lead to different (or similar) application decomposition and organization.
- Force the refinement of the concurrent programming systems so as to better support application development.
- Gain insights into the ease or difficulty of writing application code in each of the programming systems.

The common application for these experiments is Elint [KSL 86-69], a real-time, knowledge-based system for integrating pre-processed, passively acquired radar emissions from aircraft. The Elint application was implemented in three different concurrent programming systems: LAMINA, Poligon and CAGE. During the past year we completed the experiments and documented the results (KSL 88-33, KSL 88-66, KSL 88-69, and KSL 88-80).

The AIRTRAC Application

AIRTRAC (KSL 86-20) is a knowledge-based signal interpretation and information fusion system. The system attempts to identify, track, and predict the future behavior of aircraft. In particular, it attempts to recognize aircraft which might be engaged in covert activity, for example, smuggling. The inputs to AIRTRAC are periodic radar tracking system reports, a priori, filed flight plans for some aircraft, and occasional intelligence reports about suspected covert activity. AIRTRAC is designed to be sufficiently complex and realistic to adequately test various ideas about concurrent problem solving on multiprocessor machine architectures. The AIRTRAC application involves continuous input data streams, typical of real-time signal interpretation problems. Such problems often require a level of

computational power two to three orders of magnitude beyond what is currently available. Moreover, the application uses data-driven, expectation-driven and model-driven styles of reasoning. These reasoning styles encompass a wide range of paradigms in artificial intelligence.

AIRTRAC is designed as three separate modules, called Data Association, Path Association, and Path Interpretation. The Data Association module was completed two years ago (KSL 87-34), and an initial version of the Path Association module was completed one year ago (KSL 88-41). The Path Association Module is about an order of magnitude more complex than any of our previous applications in terms of both lines of code and complexity of control, reasoning and data paths. The initial Path Association Module quantitative performance experiments yielded, at best, only about an order of magnitude speedup even on large (up to 256 processors) CARE machine architectures. This speedup was about an order of magnitude less than the best speedups obtained by earlier application experiments, such as ELINT.

The primary research question that we are investigating using the Path Association Module is whether there is an inversely proportional relationship between potential speedup using parallel execution and application complexity. If this is the case, then for most knowledge-based systems of interest the speedup via parallel execution is strongly limited. Our results to date show that for the Path Association Module the speedup limitation of about one order of magnitude is intrinsic to the application. We believe that this is primarily due to two characteristics of the application: a) Its complex control paths require significant synchronization which results in large blocks of serial execution, and b) its complex reasoning requires a fairly coarse granularity of decomposition which limits the amount of achievable parallelism.

Experiments on relaxing the control synchronization did result in improved speedup. However, in all cases the quality of solution rapidly degraded as a direct function of the amount of synchronization and was, in general, unacceptable. Experiments using finer granularity problem decompositions required increasing the amount of control synchronization, and any speedup gains due to increased execution parallelism were more than offset by the increased control serialization.

Our preliminary conclusion is that for relatively simple and well-structured applications such as ELINT, two (or possibly more) orders of magnitude speedup via parallel execution is possible. However, for complex and ill-structured applications such as AIRTRAC Path Association, speedup over a well-tuned serial program by using parallel execution is probably limited, at best, to an order of magnitude. Experiments are continuing to verify this preliminary conclusion.

During the past year we completed the design and started the implementation of a parallel, dynamic classification problem solving framework which will be used to implement the AIRTRAC Platform Interpretation Module. In this framework, a classification application is

realized as a network of sub-classification nodes. Each network node is implemented as a LAMINA object, and it executes concurrently and asynchronously. Information flow along the "links" of the network is realized by message sending. Time-tagged input data is fed into the "leading edge" nodes of the network, and the resulting information is propagated through the network to its "trailing edge" nodes. These latter nodes output results to the user whenever the classification certainty factors of any entity of interest, for example, a tracked aircraft, meaningfully change.

(2.4) Knowledge Acquisition and Machine Learning

Our research in machine learning has been ramping down, due to the departure of Professors Buchanan and Rosenbloom. However, they have continued to supervise students in our laboratory during the past year.

Inductive Rule Learning

During the past year the RL induction program was extended to learn incrementally, that is from small sets of examples presented in sequence without benefit of looking at them all together. A front-end program was written to assist in the definition of RL's starting knowledge, the so-called "half-order theory". Experiments have been started on the efficacy of combining induction and explanation-based learning. In addition, we adapted RL to be a design checker in the following sense: when a design program (or designer) proposes a new design for a device, our ability to troubleshoot it later can be partly determined by RL's ability to learn troubleshooting rules for that design.

Learning by Chunking

During the past year we completed a set of experiments evaluating a representational restriction on productions that guarantees an absence of expensive chunks, with encouraging results. We have applied our domain-independent abstraction mechanism to a set of problems in two domains (mobile robot and computer configuration), and evaluated its ability to reduce problem solving time, reduce learning time, and increase the generality of the rules learned. We have run a set of experiments which evaluate the ability of rules learned in medical diagnosis to transfer to related problems (done in a reduced-size version of NEOMYCIN-SOAR). In the area of theoretical developments and system building, we have extended our work on declarative learning to allow indexing off of arbitrary features, but in the process uncovered a new issue concerned with how to deal with multiple retrieval and discrimination.

(2.5) Symbolic Simulation

During the past year we completed the implementation of a hypothesis formation system for molecular genetics. The program, called HYPGENE, generates hypotheses to account for errors in the predicted outcomes of experiments computed by the simulation system GENSIM. HYPGENE views

hypothesis formation as a design problem. It employs design operators to reason backwards from prediction errors to alter the description of the initial conditions of the experiment to ensure that the predicted outcome of the experiment matches its observed outcome. HYPGENE generates a correct set of alternative hypotheses for two different problems encountered by biologists exploring a new mechanism of gene regulation. This research is documented in a Ph.D. dissertation, entitled "Hypothesis Formation and Qualitative Reasoning in Molecular Biology," by Peter Karp (1989).

III.A.2.4. Core System Research and Development

(1) Introduction and Overview

In this section we describe progress on our core system development and work toward a distributed AIM community. As part of our charter as a national resource, we are focusing our systems activities on producing a distributed medical research environment that is both effective for our AI-related work and can be easily reproduced at other sites. In this process, we have chosen a small number of standardized hardware and software configurations to develop and support and have tried to direct our efforts at technical areas complementary to related systems activities at other sites — we are committed to importing rather than reinventing software where possible. We continue to serve as a repository of systems information and expertise for the medical AI research community, as well as the larger computer science community. We have assisted many AIM groups in establishing local computing resources and in getting access to software available in our community (for more details, see the section on Dissemination).

One of the principal thrusts of our core systems work has been to find a technically- and cost-effective replacement for the powerful and easy-to-use computing tools of the aging DEC 2060. Our criteria for the new environment have included:

- The work environment should be modern and combine graphics, pointing, and traditional keyboard modalities of interaction, as it is expected to be the primary work environment for some years to come.
- The system should support the most powerful AI research and Lisp development environment available today, possibly involving special-purpose hardware.
- The system should support small-to-medium-size AI and Lisp-based research work without requiring special hardware.
- The cost per person should be low enough as to permit putting a machine on or near every desk and to consider the system as a potential AI delivery environment.
- The system should integrate well into a heterogeneous computing environment typical of AIM research work.
- The system should be capable of editing, organizing, and printing large documents, such as theses and books.
- The system should be capable of generating and editing state-of-the-art graphics.
- The heavily-used network services provided by the 2060 (e.g., wide- and local-area network access, electronic mail transmission/routing, reception, and user access, community bboards, file service, and print spooling) must be replaced.

- The design should be incrementally extendable and augmentable as new hardware and software technologies appear and as the number of users fluctuates.
- The design should be cost-effective enough as to be replicable at smaller AIM sites that wish to benefit from our experience.
- The design should permit easy data sharing and exchange with collaborators at other sites and within Stanford University.

As detailed in our report last year, we have chosen Apple Macintosh II workstations as the general computing environment for researchers and staff, TI Explorer Lisp machines (including the microExplorer Macintosh coprocessor) as the near-term high-performance Lisp research environment, and a SUN-4 as the central network server replacement for the DEC 2060. We outlined there the myriad of tasks facing us in making the transition from the central 2060 environment to the new distributed model, including selecting and integrating tools for text processing (editing, graphics, formatting, and bibliographic references), presentation graphics, printing, help facilities and distributed information access, interpersonal communication tools (EMail and BBoards), file management (storage, access, backup, and archiving), and system building tools (languages, development environments, and integration tools).

However, rather than the carefully orchestrated transition plan we had proposed and received approval from Council to implement (see last year's report for a summary), we were obliged to undertake a much more precipitous transition because of the severe funding constraints resulting from an 11% cut in our NIH/DRR grant award last year. There were two immediate consequences of this large budget cut: a) reducing our systems staff by two people (one layoff and one through attrition) and b) taking the DEC 2060 off of contract maintenance early in the grant year, thereby forcing us to close it down for routine use. These steps have had other impacts in slowing our work toward the long-term research goals we set out to achieve, both in forcing us to devote full energy to the 2060-to-SUN-4 transition approximately a year before we expected to be ready for it and in reducing the level of effort of the remaining staff for work on these difficult problems.

Because of the necessary preoccupation of most of our staff with this premature transition this past year, we were not able to convene the visiting advisory group as was recommended by BRTP to help guide our long-term research efforts. As we finally close out the 2060 chapter this summer, we will plan to assemble such a group in the early fall (September or October) to reassess our plans for the coming two years. In spite of all this unplanned redirection of our energies, we have made substantial progress this past year as summarized in the following sections.

(2) The Phase-Out of the DECSys-20

The contract maintenance cost of the DEC 2060 was about \$70,000 per year and we could not afford to continue this coverage in light of the large budget cut. Since this old-technology machine would become unreliable without regular maintenance, this forced us to transfer nearly all of our AIM community usage to the SUN-4/280 in October and November of 1988. The DECSys-20 had been our major computer resource since February of 1983. As this machine, in turn, had replaced a DEC KI-TENEX system in use for nine years, our conversion to the UNIX based SUN-4/280 represented a major departure from a long-held approach to computing.

While some of our users already had experience with the UNIX operating system and hence had an easy time of adapting to our SUN-4 system, most others had a long history of using only the TOPS-20 operating system or its predecessor, TENEX. For these users, converting to the use of UNIX was a major obstacle. The more mnemonic command syntax of TOPS-20, together with the various command completion and ever-ready "help" features are not available in the basic UNIX system. Even the popular "man" feature for the on-line reading of UNIX manual pages provides little in the way of tutorial guidance.

A significant and urgent effort went into developing a *UNIX Users Guide for TOPS-20 Users* which has provided substantial help in navigating through the most common of commands. But, it is clear that despite its overwhelming popularity, the UNIX system fails to provide many of the user-friendly features available for many years in the TENEX and TOPS-20 environments.

The mechanics of transferring about 400 user accounts was a major but relatively straightforward task. We were fortunate in having available a program from Rand Corp. (via Rutgers University) which provides for the reading TOPS-20 Dumper tapes under UNIX. Most of the immediately-needed working files from the 2060 system were dumped to tape and loaded into the SUN-4 using this program. In addition, the Ethernet system connecting the two machines facilitated the task of transferring files and ensuring on-going access to the files remaining on the 2060. Most of this transfer was done during a four week period of intensive work.

As complicated as the transfer of the users' files was the handling of transferring the "SUMEX-AIM" name from the 2060 to the SUN-4 (including coordinating updates to all of the domain servers and host tables around the Internet), mail distribution issues, and providing continuation of BBoard facilities. Another month of planning and implementing these aspects of the transition were required, including a number of false starts because of problems in the ARPANET Network Information Center in timing the Internet name and address changes. Continuous and nearly compatible AIM community mail services were maintained through the transition by installing the Columbia University MM-C mail program on the SUN-4. This program closely duplicates the functions of the TOPS-20 COMAND JSYS

under UNIX and presents the user with a mail reader/composer interface very similar to that of TOPS-20 MM — the system that had been used by the AIM community for 10 years. At the time of our SUN-4 transition, MM-C was still in its early stages of being released and required intensive work to track down numerous bugs which were uncovered by the heavy use in our community. Nevertheless, this system, coupled with a UNIX version of the EMACS text editor, called GNUEMACS, provided a familiar setting for the most common computing functions used by AIM community members.

Once this program was in place, the 2060 mail files were passed through a preprocessor to make them UNIX-compatible. Next, we forwarded all mail directed to the old SUMEX-AIM (the 2060) to the SUN-4 (the new SUMEX-AIM). This allowed mail to be accessed during the transition. Once the transition was completed, we changed the name of the 2060 to be SUMEX-2060, and renamed the SUN-4 SUMEX-AIM. From this point onward the 2060 was no longer used for mail access. In the succeeding months, we added bulletin board functionality to MM-C so that, from the user's perspective, mail access was nearly identical to the former system.

Part of our original plan for conversion from the DECSystem-20 to the SUN-4 included moving the 2060 ARPANET interface to the SUN-4. This plan was initiated in the fall of 1988 but, before SUN Microsystems could deliver the required hardware/software interface package (and BBN could provide a replacement IMP interface) major changes in ARPANET service were being implemented. As a result, the ARPANET connection for the SUN-4 was cancelled and our Internet access is now implemented through the Bay Area Regional Research Network (BARRNet) and the NSFNet (See Wide Area Network Developments).

Another major issue in the 2060-to-SUN-4 transition has been the need to provide our users with continued access to their large collection of archived files and to a set of permanent annual backup dumps (done January of each year) which have been collected and maintained since 1975. Early tapes are in the BSYS Archive format or TENEX Dumper format. Later ones are either TOPS-20 Archive format or TOPS-20 Dumper format.

This has required very careful planning as the directory information for these tapes resides in Archive directory files (for TENEX) and the on-line File Descriptor Blocks (FDB's) of the TOPS-20 file system. These two sets of information must be converted to simple UNIX-compatible text files to provide users with continued facilities to review and access their collections of archived files. This work has been a major undertaking and is still in progress. (See the Section on File Access, Back-up, and Archiving for more on this topic).

Our past commitment to the use of Ethernet TIPS for the connection of users to hosts proved its value in the system changes described above. Little concern had to be given to moving collections of terminal "hard-lines" from one piece of equipment to the other. Likewise, with dial-in modems attached to TIPS, users could easily select the new system when establishing

connections. More recently, we have been able to restore telephone dial-out service by upgrading our EtherTIP systems. Now users can connect to dial-out ports on the EtherTIP from their workstation directly (e.g., Macintosh) or from the SUN-4 to make external modem connections. Our X.25 service line from TELENET Inc. also is attached to an X.25/Ethernet gateway which provided users immediate access to the new SUN-4 system from TELENET (see Wide Area Networking).

Although we are still missing a number of features in the new system (e.g., adequate community and project directory and file management structures, usage accounting, a WHOIS data base, and a fully-operational archiving system), the physical aspects of transferring to the SUN-4 are nearly complete. During this transition, the 2060 has been maintained only on a "time & materials" basis as absolutely necessary to retain adequate access. Numerous parts of the system have failed in the intervening 9 months, including several memory banks, disk problems, and a backplane problem. The system is now quite unreliable but not worth repairing as its resale value is nearly \$0. We hope to keep it running through this summer in order to complete the file system transfer to UNIX and to tie off access to old 2060-based software. At that time, the 2060 will be shut down permanently, ending an important phase in the history of SUMEX-AIM and the AIM community.

(3) The New SUN-Based SUMEX-AIM Resource

The SUN-4/280 central server for the SUMEX-AIM resource was acquired in 1988, is configured with 32 megabytes of memory and 1.8 gigabytes of disk storage, and runs under the UNIX operating system (see Figure 3). The primary function of this machine is to provide, in place of the 2060, wide- and local-area network access; electronic mail transmission/routing, reception, and user access; community bboards; file service; and print spooling for the AIM community. Such UNIX servers, running on modern high-performance hardware like the SUN SPARC chip or other RISC chips, are relatively inexpensive and fast and are easily obtainable by other groups in the AIM community.

We also operate a complementary SUN-3/180 machine, acquired in 1987, which is named KNIFE (KSL Network Interface and File server Environment) and which provides additional file storage and access facilities for the community (see Figure 4). Both SUN servers run the same version of the SUN Operating System (SunOS 4.0), providing the expected efficiency advantages from uniformity — the KNIFE server had been running the earlier release 3 operating system and was only recently updated to release 4.0.

A third VAX 11/750-based file server (SAFE) is being phased out in deference to the more cost-effective SUN systems and its disks will be reintegrated with SUMEX-AIM and KNIFE.

These servers provide distributed computing services, including NFS (Network File Access), to the various personal computers and workstations in the AIM community. SUMEX-AIM is the main EMail machine and provides individual mail services, as well as network distribution lists and bulletin boards. KNIFE, on the other hand, is more committed to network and distributed file service development and support. Having two such different yet similar environments has provided good flexibility in offering various functionalities, while easing the administration of the distributed systems. We anticipate exploiting this duality in the coming year, specifically in the area of file backup and archiving.

(3.1) File Access and Management

A stable, efficient mechanism for storing and organizing data is central to any computing environment, and is one of the most challenging issues in the move to distributed, workstation-based computing. It is necessary to provide standard services, such as file backup, archiving, a flexible and intuitive naming facility, and data interchange services (e.g., file transfer). Also, as the amount of data being manipulated grows, it becomes more and more important to have powerful tools for managing hierarchies of files.

UNIX has many of the needed facilities, e.g., backup, long names, hierarchical directory structure, some file property attributes, data conversion, and limited archival tools. However, while general issues of networking, remote memory paging services, and flexible file access have received considerable attention in both the academic and commercial development of file servers, there has been only slow development of other critical operational tools. For instance, the much-used file archiving system of the DEC 2060 (sometimes called off-line cataloged storage) has no analog service in "standard" UNIX systems. Perhaps this is the result of UNIX having its origin in the small computer world where the number of users and volume of data has traditionally been quite low. To ensure continued availability of archiving services in our transition from the 2060, we have worked on adapting a commercial system developed by UniTech to allow users to manage large file collections by moving files not needed on-line to and from off-line tape storage. This system also maintains a historical archive of files. See the section on the "SUMEX Perpetual Archive System" below.

We have had a well structured organization for managing disk usage and accounting on our 2060 system and we plan to duplicate some of these features on the SUN-4. A UNIX accounting system will be put in place to allow cost accounting and provide a quota enforcement capability on the distributed file servers. UNIX has always been weak in the management of large-scale file backup and with the advent of disks of near gigabyte size, it is clear that a better organized approach is essential. In support of the long-term goals of the distributed community, we have been reviewing more advanced data storage methods. Optical disk systems are attractive but have

not progressed as quickly as many had predicted. However, it seems likely that this sort of equipment would be ideally suited to satisfying our requirements for the archiving of files. Helical-scan magnetic tape equipment might also serve this function well. However, in both cases the absence of industry standards introduces some level of risk when planning to use these types of data storage equipment. We plan to continue our investigations of these technologies this coming year and to make a decision.

CAP, the public domain Columbia AppleTalk Package, is installed on both servers. Within its facilities, the AppleTalk/UNIX File Service package (AUFS) provides icon-based file support on a UNIX server so that Macintosh workstations can connect and see representations of their UNIX files in the same format as with the file tools on the native Macintosh. File transfers are invoked by moving icons around the Macintosh screen just as if the UNIX server were an extension to the internal Mac desk top and disk. This system operates either through Kinetics FastPath AppleTalk/Ethernet gateways or through pure Ethernet connections.

(3.2) The SUMEX Perpetual Archive System

Overview

The SUMEX Perpetual Archive System is an integrated set of system software and operator procedures which allow individual SUMEX users to create and manage perpetual magnetic tape archives of their files and directories. Features include:

- A command which requests the system operator to move the contents of specific files or directories to the magnetic tape archive. Note that the files are first moved to an intermediate on-line repository with other files awaiting archiving, so unless a user specifically requests that the contents of a file be retained on-line, the disk space used by the archived files will be immediately removed from their on-line disk quota.
- A command to interrogate an on-line catalog of archived files.
- A command to request the retrieval of archived files. In the case where a file of the same name is entered into the archive more than once, the user may specify a date such that the version of the file entered into the archive just prior to that date will be retrieved.

Historical Perspective

The current implementation of the SUMEX Perpetual Archive System is actually the third such implementation at SUMEX. The first was based on the BBN TENEX operating system, and utilized the BSYS magnetic tape backup software package written by Smokey Wallace at the SRI Augmentation Research Center in 1974. This BSYS-based system was in place from 1975 until 1984, when SUMEX was upgraded from a dual-processor DEC KI-10 system to a DECSYSTEM 2060 running the DEC TOPS-20 operating system — a direct descendant of BBN TENEX. By that time,

DEC had integrated into TOPS-20 an archiving capability that used the regular magnetic tape backup program, called DUMPER, also originally from BBN. The DUMPER based system was place from 1984 to 1988, when SUMEX was once again upgraded, this time to a more distributed system based on a SUN-4/280 network and file server, running UNIX. The current system utilizes a third-party magnetic tape backup software package called UBACKUP from UniTech Software Inc., which in turn utilizes the standard UNIX "tar" magnetic tape backup program.

As the SUMEX computing facility has changed operating systems, first from TENEX to TOPS-20, and then from TOPS-20 to UNIX, facilities have always been provided to retrieve files from the archives of the previous system — hence the basis for calling this archive system *perpetual*.

Implementation:

Normally, when a UNIX account is created for a user, he is assigned a home directory on a file system (for example /a/user) and is assigned an allocation of disk space on that file system called the *quota*. At SUMEX, an additional directory, called the user's *archive directory*, is created on another file system, in this case /var/archive/user. Once the archive directory is created, any files moved into it will automatically be moved to the magnetic tape archives the next time the operator runs the archive utility. The frequency of moving files from the archive directories to tape is a function of the total amount of space allocated to all archive directories, but is guaranteed to be at least once a week (on Sundays), and not more than once a day.

Commands

The archive and retrieve commands have a similar format, and can be thought of as commands that move files and/or directories between a user's home directory, say /a/user, and archive directory, say /var/archive/user.

archive [-d] [-h] **pathname...** [-as **pathname2**]

retrieve [-h] **pathname...** [-as **pathname2**]

Key features of the *archive* and *retrieve* commands include:

- Multiple arguments and options are allowed, in any order, and wild cards will be expanded by the shell as usual.
- The *-d* switch specifies that files requested for archiving are to be deleted from the system afterward.
- The *-h* switch specifies that the specified files and/or directories are to be archived from or restored to the *top* level of the user's archive directory. For directories, the structure below the specified directory is preserved.
- The *-as* switch allows the user to specify a different name under which the file or directory is to be archived or restored.
- Either absolute or relative *pathnames* may be supplied, i.e. it doesn't matter where the user is connected at the time of issuing the archive

command (although the examples below all assume the user is connected to the home directory).

- In the case of directories, all subdirectories of that directory will be recursively archived.
- If a filename conflict occurs because the user requests a file by the same name be archived before the previous request has been processed, archive will warn the user and ask for confirmation.
- If a filename conflict occurs because the user requests a file by the same name be retrieved, and a file by that name already exists, retrieve will warn the user and ask for confirmation.
- When requesting the archive of files and/or directories outside the user's home directory, the standard UNIX protection scheme applies.
- Deletion of the archived files is left to the user.

Examples

In the following examples, it is assumed that the user is logged into his directory (username) and is connected there:

Archive request	Archived as
archive .login	/var/archive/username/.login
archive mail/mail.sent	/var/archive/username/mail/mail.sent
archive /a/username/mail/mail.sent	/var/archive/username/mail/mail.sent
archive -h mail/mail.sent	/var/archive/username/mail.sent

(3.3) Printing Services

Laser printers have long ago become essential components of the work environment of the SUMEX-AIM community with applications ranging from scientific publications to hardcopy graphics output for ONCOCIN chemotherapy protocol patient charts. We have done much systems work to integrate laser printers into the SUMEX network environment so they would be routinely accessible from hosts and workstations alike. This expertise has been widely shared with other user groups in the AIM community and beyond.

SUMEX operates 4 medium-speed (8-20 pages per minute) Imagen laser printers, 6 low-speed (~3 ppm) Apple laser printers, 1 low-speed (~3 ppm) Xerox laser printer, and 1 low-speed (~1 ppm) Apple color dot-matrix printer.

All of the Imagen printers incorporate an emulator for a line printer, a Tektronix plotter, and a typesetter (using the *Impress* language). Additionally, the two Imagen 3320 printers implement the *PostScript* typesetter language (also implemented by the Apple LaserWriters) required

for printing Macintosh documents. The Xerox printer (an 8046) interprets the *InterPress* typesetter language. The Apple *ImageWriter LQ* is a low-cost dot-matrix printer that the Macintosh's *Quickdraw* printer driver is capable of using for color printout.

IMAGEN host software is installed on both the SUMEX-AIM and KNIFE servers, providing print spooling services for client workstations to any of the several Imagen Printers in our computing environment. Since two of the Imagen printers (3320's) provide UltraScript support (a PostScript-like document description language), we have connected the CAP and IMAGEN software together such that Macintosh users can spool print jobs to the IMAGEN printers. These printers appear in the Macintosh *Chooser* the same as any local LaserWriter.

Since the NeXT computers print with the PostScript language, we didn't acquire any additional printers to support these workstations.

In total, the laser printers printed about half a million pages of output during the year. Most of the printout was simple text (primarily electronic mail listings) and documents formatted on the Macintosh — technical papers, drawings, program listings, and screen dumps. As was our intention, only a very small number of charts was printed on the *ImageWriter LQ* — complicated charts and graphs that benefitted from the use of color to distinguish otherwise indistinct graphic or data elements.

(4) Electronic Mail

Electronic mail continues as a primary means of communication for the widely spread SUMEX-AIM community. As reported last year, the advent of workstations has forced a significant rethinking of the mechanisms employed to manage such mail in order to ensure reliable access, to make user addressing understandable and manageable, and to facilitate keeping the mail software distributed to workstations as simple, stable, and maintainable as possible. We are following a strategy of having a shared mail server machine which handles mail transactions with mail clients running on individual user workstations. The mail server can be used from clients at arbitrary locations, allowing users to read mail across campus, town, or country.

The mail server acts as an interface among users, data storage, and other mailers. Users employ a Mail Access Protocol (MAP) to retrieve messages, access and change properties of messages, manage mailboxes, and send mail. This protocol should be simple enough to implement on relatively inexpensive machines so that mail can be easily read remotely. This is distinct from some previous approaches since the mail access protocol is used for *all* message manipulations, insulating the user client from all knowledge of how the mail is actually stored on the server. This means the the mail server can utilize whatever data storage and access methods are most efficient to organize the mail on a particular server system.

Thus, a user sitting in front of his personal workstation can read mail from any system on which he has an account and on which such a mail server is running. This has several advantages to the usual scheme of TELNETing to such a host and then reading the mail on the host, and receiving the text in a byte stream over the network in use. Given a MAP designed for efficiently accessing mail on such a network, the effective bandwidth for text transfer can be maximized in two ways. First, when mail is accessed, a compact representation of the users new mail can be sent for him to peruse and act upon. This information is essentially an envelope containing addressees and the subject of the message — the text of the message is sent only when the user decides to read it. Efficient clients can read ahead and cache envelopes while the user is perusing information and composing messages or when the client is notified of the arrival of new mail. This maximizes the users' *perceived* bandwidth since the protocol minimizes the delays between the receipt of useful information. Secondly, the MAP server knows it is dealing with mail and can maximize network bandwidth by maximizing the data in each packet it sends in reply to a MAP request. In our servers and clients we utilize these techniques whenever possible.

The first prototype Interim Mail Access Protocol (IMAP) was designed with this in mind. As noted in our previous report, the prototype IMAP server on the DEC-20 provided the foundation for the IMAP protocol version 2 (IMAP2). During this reporting period the IMAP2 protocol description was published as an ARPANET Internet working group Request for Comments¹, making the IMAP2 specifications available to the general ARPANET community. This has resulted in IMAP2 clients being developed elsewhere. In fact, such a client has been implemented in Japan by NTT to run on their ELIS lisp machine. The implementer regularly reads his mail in the U. S. from Japan using this client.

We completed several tasks noted in last year's report — the DEC-20 server and Xerox Lisp client were upgraded to IMAP2; we implemented an IMAP2 server for UNIX; and demonstrated a prototype client in Common Lisp for the Texas Instruments Explorer. In addition to finishing this work in progress, the main thrust of our effort this year was to write a mail client for the Macintosh. The Macintosh client will be in alpha test by early summer 1989. In addition, a UNIX workstation mail client, called MS, using the Columbia MM CCMD package, has been implemented with an MM-like command interface. The MS client was initially developed at SUMEX as a debugging tool since it uses a command line user interface rather than one driven by a mouse, menus and windows. MS also runs under MS/DOS on IBM PC's, and on NeXT machines. A NeXT client with a graphics interface is also well on its way, and is being developed at the University of Washington.

¹ Crispin, M. R. "Interactive Mail Access Protocol - Version 2." Internet Working Group Request for Comments No. 1064, Stanford University, July 1988.

(4.1) Macintosh client - MacMM

When we closely examined the design of a client, it became clear that a large portion of the code was machine-independent, and should be written in such a way that it could be ported to non-Macintosh systems. A client comprises a mail reading module, mail composition module, IMAP2 protocol module, SMTP (Simple Mail Transport Protocol) module, and a TCP-IP network communication module. The first two of these require extensive user interfaces. When we began this project, we decided to write it in C because of its availability on most machines. At that time Apple was alpha-testing an TCP-IP device driver, MacTCP, for the Macintosh, but a stream or socket interface to this code to facilitate simultaneous use by multiple applications was not available and had to be developed. It is important to note that MacTCP supports multiple active applications using TCP-IP simultaneously, and this is very important in our environment. Since the last three modules mentioned above could be written and debugged under UNIX, we started the project on two fronts.

A simple line editor reader/composer user-interface was written to run under UNIX. This was initially considered to be "throw-away" code, but as mentioned above, has evolved into three separate IMAP2 clients. The IMAP2 and SMTP modules were then written in C to be machine independent, as was a small interface module to the operating system-dependent TCP-IP package. Thus for each new client, one was required to write the machine dependent calls to the resident TCP-IP code, and the mail reader/composer modules. At the same time we undertook a project to write a high level stream interface to the Apple MacTCP TCP-IP driver that was in alpha release.

These two projects proceeded in parallel and were finished at about the same time. The UNIX-based client, MS, was then ported to the Macintosh in the simple line editor mode, interfaced to the TCP-IP code, and was used to debug the IMAP2 module. At this point, we began to write the graphics-oriented mail reader user-interface.

(4.2) Mail Reader User-Interface

We began designing the user interface (i.e. client) in mid-September 1988. Although we knew what commands we would have to implement (from our earlier work on the Xerox Lisp machine client), we had to maintain the "look and feel" of the Macintosh for each of these commands. Additionally we wanted to be able to run other applications concurrently with the mail system and to bounce back and forth between these applications. This implied that the data structures maintained by the client be kept to a minimum to avoid running out of memory. We also had to get a working understanding of the Macintosh system calls before we could start coding.

We began the actual coding in late October. The first task was to implement a status window to inform the user of various events (such as notifying the

user that there are new messages) and also as a place to output debugging information. This is an informational window only. We also had to implement the basic menu system. Almost every application has at least three menus (the Apple, File, and Edit menus) and the mail system is no exception. The File menu contains global commands such as "Open Mailbox" and "Compose Message". The Edit menu allows the user to Copy, Cut (copy and delete), and Paste (insert) selected text. Although the Edit menu is primarily used in composing a message, users can also manipulate text in the read and status windows.

The next step was to implement a scrollable message header browser window which has a one-line descriptor (message number, date, sender, subject, and size of message in characters) of each message in the user's mailbox. This is the heart of the mail system in that the user must select messages in this window before anything can be done to these messages. For example, to read a message the user must first select the message to be read. To select a message the user just points the mouse to the message descriptor and clicks it. There are two menus associated with the browser window; one that operates on the selected messages and another that affects the entire mailbox. The message menu allows the user to Read, Flag, Delete, etc. the selected messages.

There is also a "Select" menu item that allows users to automatically select messages that have certain properties. The user specifies these properties by buttoning fields in a "dialog" box. The mailbox menu allows users to Expunge deleted messages, Check for new messages, and Zoom selected messages (replace the browser display with only those messages that the user has selected). As of this writing, the Zoom and Print (to hardcopy messages) commands have not been fully implemented.

Next we implemented Read windows and the associated Read menu. The Read menu allows the user to operate on the current message in the Read window. The user can simply continue on and read the next selected message or perform some action on the current message. These actions include Answer, Delete, Flag, etc.

The entire mail system is menu/window driven. By this we mean that the active window determines which menus are enabled and the contents of the active window determine which menu items (in the enabled menus) are enabled. For example, if the message browser window is active, then only the browser menus are enabled.

Additionally if there are no messages selected, then none of the items in the message menu are enabled (with the exception of the Select command) since these commands only operate on selected messages. This approach eliminates confusion since the user can see at a glance which commands are allowed and which are not. The initial release of Macintosh MM will only allow one open mailbox at a time and one read window. The code was designed to allow multiple mailboxes and read windows so this restriction will be removed soon.

(4.3) The Mail Composition User Interface

In late March 1989, we began the mail composition user interface. This interface is also mouse/menu driven and supports simultaneously opened composition windows. Each such window has two independent panes; the top pane is for the composition of the mail header information, and the bottom for the text of the message itself. Each pane has independent scrolling and fonts. The mail header can be checked for the correctness of RFC822 syntax at anytime, and any addresses in error are flagged by both an error message "alert" and cursor pointing to the portion of the text that contains the error. The mail header can be freely edited, much like one would do in Emacs, except that the field headers, e. g., "From: ", are protected in the sense that they cannot be deleted. Both panes support the usual Macintosh text editing features such as cut, copy and paste. At the time of this publication, one can compose and send messages, save and restore drafts of messages, and the composition user-interface is essentially completed. Also, one can tailor the mail composition environment to ones own needs.

For this customization to be accomplished, each Macintosh has a composition initialization file. Using this file one can set a personal name, default font, font size and font face, i. e., plain text, **boldface** or *italics*.. Also, a system administrator can set the default SMTP server name list, and the host from which the mail appears to be sent, and to whom the replies may be sent.

Several features need to be added and are currently in progress. Among these are mail forwarding, message reply-to and answering, and the queuing of unsent mail and background mailing of this same mail. It is expected that this list may grow by user demand during its alpha and beta testing phases.

(4.4) Texas Instruments Explorer Client

The IMAP2 client for the TI Explorer system using Common Lisp mentioned in last year's report is still being developed by not as intensively as the Macintosh client because of staff limitations, and the fact that the AIM "market" for the Explorer system is significantly smaller. Nonetheless, substantial progress has been made on this client by James Rice. The TI Explorer IMAP Client is in a partial state of completion. The basic functionality for sending, receiving and replying to messages is defined, as is support for BBoards. The major areas where more work is needed to make it usable are in the command handler; significant modification of the original prototype implementation is needed, it still has a number of bugs, and the interface with the Zmacs text editor works but will probably need modification. We estimate that an additional month of work will be required before the system is ready for alpha-test.

(4.5) DEC-20 IMAP2 Server

The current version of the DEC-20 server is a complete and robust implementation of IMAP2 and we do not anticipate any further work on it.

(4.6) UNIX IMAP2 Server

UNIX is becoming more and more widely adopted in the AIM community and in our distributed mail system we will rely heavily on UNIX engines as servers. As summarized in last year's report, we have written a UNIX IMAP2 Server (UIMS) for remote mail access. UIMS handles the full complement of IMAP2 commands, and has been vigorously tested for the past year.

This past year, we benchmarked UIMS against Columbia MMC on a SUN-3/180 running SUN OS 4.0.1 to compare speeds for text searches of reasonably large mail files. This was done by modifying the MMC code to calibrate searches. The UIMS already has calibration incorporated into its code. Note that this compares wall clock search time between UIMS and MMC, and in both cases file I/O is factored out. It is interesting to note that making these measurements for UIMS was tricky. This is because, unlike MMC which reads the entire mail file at initialization, UIMS is very careful to minimize its working set size in order to reduce paging overhead and UNIX scheduler penalties for programs with large working sets.

UIMS only keeps messages in memory if the search succeeds, or if the message number is within 40 of the number of messages in the mail file, or if the user has "shown interest" in the message prior to the search. So, two searches were done in UIMS. The first was for the string "a" which coerced all of the data into memory, and the second was for the string "ImNotInAnyMessage%%%". Thus, the second search is independent of file I/O.

The results of these measurements for various mail file sizes are summarized in the following table:

Mail File Size	MMC Search Time	UIMS Search Time
2.5 megabytes	44 secs	11 secs
1.2 megabytes	16 secs	2 secs
0.5 megabytes	4 secs	1 secs

Table comparing EMail text search times for MMC and UNIX IMAP

Note that 0.5 MB is approximately the typical mail file size on SUMEX-AIM, ignoring large users and the very large BBoard files. The data illustrate the efficiency that can be gained by accessing mail from a special purpose server which is optimized for access performance, independent of the user interface code which executes separately on the user's client workstation.

The principal development effort on the UIMS this past reporting period was the addition of code to allow flexible access to read-only bulletin boards. In addition, we fixed a number of bugs reported by client developers.

We believe that accessing mail in this client/server model will prove to be more efficient than the usual mode of using mail reading programs on the mail server itself, or using network file system protocols to remotely access mail (i.e., down-load the entire mail file to the workstation. In the former case, the operating system overhead for running a job under UNIX has much more impact on system resources, than does an UIMS server process. The latter has no shell process for executing commands, and was written carefully to minimize system resource usage. The example of the search comparison above clearly points out this difference. Secondly, network file system protocols, like SUN Microsystem's NFS, show a drastic decrease in performance as the distance between the server and client host increases. This is even true on a local area network, where a client and host are separated by one or more gateways. This is not evident with our mail client/server model because IMAP2 was designed to be transactional and minimize the impact of a server and client being on separate subnets of a local area network, or even being separated widely on the Internet.

(4.7) Transition Strategy and Plan

The transition strategy plan described in last year's report is well underway. The AIM community currently reads mail on the SUMEX-AIM SUN-4 using the Columbia University MMC system which closely emulates the earlier TOPS-20 MM interface. UIMS understands the MMC mail storage format and we can access the same mail on one of the several personal workstations mentioned above. By the end of this summer, we expect most users will be reading their mail using an IMAP client on Mac II or Explorer workstations during the day, and in the evening, they will use a TELNET connection to SUMEX-AIM from a terminal emulator to process mail with MMC.

It is again important to mention that Columbia University distributes MM-C with the statement "Permission is granted to any individual or institution to use, copy, or redistribute this software so long as it is not sold for profit, and provided this copyright notice is retained". This makes MM-C an ideal mail reading program for the SUMEX-AIM community since our move away from the DEC-20 toward a distributed environment, since this interim mail-related software can be made widely available to the national community.

(5) Lisp Systems

(5.1) Standards

In a heterogeneous computing environment, such as AI research inevitably involves, the issue of cross-system compatibility is a central one. Users of various machines want to be able to share software, as well as be able to use various machines with a minimum of overhead in learning the operating

procedures and programming languages of new systems. Thus, it is crucial to specify and propagate powerful, flexible standards for various aspects of the computing environment so that it is possible to transfer both skills and information among machines.

In order to improve the inter-machine compatibility of our software, we have been encouraging all users to use the Common Lisp programming language¹, as well as pressing vendors to provide more complete and efficient implementations of this language. We have already served as beta test sites for Xerox, Texas Instruments, and Lucid Common Lisp implementations.

The Common Lisp language, however, is only a subset of the software needed for our research. Research projects need higher-level powerful facilities, such as an object-oriented programming system, sophisticated debugging and error handling tools, portable window systems, and better graphics interface development tools. Therefore we have been supporting and following the development of evolving standards such as the Common Lisp Object System (CLOS), Common Lisp X Window system (CLX), and proposals for higher level User Environments (e.g., CLUE) via membership in the electronic discussion groups and specific technical contributions.

(5.2) Lisp System Performance

One of the key issues in selecting the systems for our distributed computing environment was the performance of Common Lisp. In order to assist us in evaluating the performance of Lisp systems, we undertook an informal survey of Common Lisp environments using two KSL AI software packages, SOAR and BB1. The results have been compiled into a KSL Technical Report presented in Appendix B. In this survey we focused on execution speed for simulated runs of the test programs and for compilation of the test programs as measures of performance. However, we emphasize that execution speed benchmarks are only one aspect of the system performance evaluation, especially for Lisp systems. Other crucial issues like programming and usage environments, compatibility with other systems, ability to handle "large" problems, and cost must also be considered.

Both SOAR and BB1 were chosen because they are implemented in pure Common Lisp, making them extremely portable. SOAR is a heuristic search based general problem solving architecture developed by Paul Rosenbloom et al.² and BB1 is a blackboard problem solving architecture developed by Barbara Hayes-Roth³. Neither of these systems is an intensive user of

¹ Steele, G. L., Jr. *Common Lisp - The Language*. Digital Press, Burlington, MA, 1984.

² Laird, J. E., Newell, A., and Rosenbloom, P. S. "SOAR: An Architecture for General Intelligence." *AI Journal*, 33(1):1-64, 1987.

³ Hayes-Roth, B. "A Blackboard Architecture for Control." *AI Journal*, 26:251-321, 1985.

numeric computation. They were initially developed in environments other than those tested and no attempt was made to optimize their performance for any of these tests.

The workstation systems to be tested were chosen based on their availability as well as projected applicability in AIM community environments. Since we were interested in "real world" results, we ran the tests on each machine in what seemed to be its standard operating mode. The machines tested include:

- SUN 3/260 with Lucid Lisp1
- SUN 3/60 with Lucid Lisp
- Compaq 386 with Lucid Lisp
- Compaq 386 portable with Lucid Lisp
- SUN 4/260 with Lucid Lisp
- SUN 4/280 with Lucid Lisp
- DEC MicroVax II with VaxLisp
- DEC MicroVax III with VaxLisp
- SUN 3/75 with Franz Extended Common Lisp
- Texas Instruments Explorer I
- Texas Instruments Explorer II
- Texas Instruments Explorer II Plus
- SUN 4/280 with Franz Allegro Common Lisp
- Hewlett Packard 9000/350
- SUN 3/75 with Kyoto Common Lisp
- SUN 3/75 with Lucid Lisp
- Apple Macintosh II with Allegro Common Lisp
- Symbolics MacIvory
- Texas Instruments microExplorer
- IBM RT/APC with Lucid Lisp
- Symbolics 3645
- Xerox 1186

A large variation was observed between the ranking of systems when running the SOAR test and when running the BB1 test. This leads us to conclude that while these experimental results, and ones like them, can be used to

Hayes-Roth, B., and Hewett, M. BB1: An Implementation of the Blackboard Control Architecture. In *Blackboard Systems*, Englemore, and Morgan editor, Pages 297-313. Addison-Wesley, Palo Alto, CA, 1988.

class machines together roughly, it is impossible to use such a set of benchmarks to decide in advance how a given application will perform on a given system. There is no substitute for actually running the program on the systems in question. However, on the basis of these data, the systems tested may be ranked as follows:

- Very Fast (≤ 0.50 anr — averaged normalized run time): TI Explorer II Plus (Exp2+), TI Explorer II (Exp2), and SUN 4 with Lucid Lisp (4/280 and 4/260)
- Fast (> 0.50 anr, ≤ 1.00 anr): TI microExplorer (mX), Compaq 386 (386), SUN 4 with Franz Lisp (F-4/280), Compaq 386 portable (386T), SUN 3/260 (3/260), IMB RT/APC (RT), and SUN 3/60
- Medium (> 1.00 anr, ≤ 1.50 anr): Symbolics 3645 (Sym), SUN 3/75 with Lucid Lisp (L-3/75), HP 9000/350 (HP), TI Explorer I (Exp1), and DEC MicroVax III (DEC-III)
- Slow (> 1.50 anr, ≤ 2.50 anr): Symbolics MacIvory (Maci), SUN 3/75 with Kyoto Common Lisp (K-3/75), and SUN 3/75 with old Franz Extended Common Lisp (E-3/75)
- Very Slow (> 2.50 anr): Apple Macintosh II with Allegro Lisp (Mac2), DEC MicroVax II (DEC-II), and Xerox 1186 (XCL),

The data were normalized by dividing individual results by the average of the results for all the tested implementations.

As new Common Lisp implementations become available to us we plan to revise the report to include measurements of these systems. We are currently integrating measurements for Symbolics MacIvory running Genera 7.4 and Symbolics 3650 running Genera 7.2. We are working on getting measurements for Symbolics XL400, DEC MIPS, and Cray 2, as well as other mainframe systems.

(5.3) Lisp Programming Environments

Even though performance gaps between microprogrammed Lisp systems and stock workstation implementations are narrowing, there still remains a significant difference in the quality of the development environments. The power of the development tools and environment is what has been the primary strength of Lisp machines, allowing rapid design, implementation, and debugging of complex programs. We believe the key to good development tools is integration, both in terms of consistency of interface, and in the ability to move seamlessly from tool to tool, carrying along appropriate data and state information. These qualities must be manifest in any AIM research computing system.

Over the years, KSL and AIM community AI systems have been implemented predominantly in the InterLisp, MACLisp, ZetaLisp, and more recently, Common Lisp dialects. Beginning in the early 1980's, our work moved from mainframe Lisp environments to workstation environments for many

reasons, principally involving powerful tools for system development and debugging and graphical interfaces. Commercial versions of these tools, that evolved over many years in the Xerox D-Machine, Symbolics 36xx, LMI, and TI Explorer systems, have become an indispensable part of our work environment. Newer Lisp systems for workstations not specifically developed for Lisp have lacked many important features of these environments.

Thus, in light of the runtime performance advances of stock workstations, we have attempted to summarize the key features of the Lisp machine environments that would be needed in stock machine implementations in order to make them attractive in a development setting (the first draft of this specification was included in last year's report). Unfortunately we have been unable to refine this draft specification over the past year due to lack of resources. We hope to better address the issue of high-quality development environments on a variety of platforms in the coming year.

(6) Workstation System Environments

(6.1) Macintosh II Workstations

Hardware and Software Environments

The installation of the Macintosh II workstations and ancillary networks, printers, and software went mostly according to plan, and most have been in productive use for over a year now.

The inexpensive Rodime winchester disks have proven quite reliable so far. The few units that have failed were replaced under warranty. A greater number of Moniterm displays exhibited a tendency to fail from overheating, with annoying but not problematic frequency.

We are using Microsoft's *Word* for word processing, *PowerPoint* for presentations, *Excel* for accounting, and Blue Sky's *TEXTures* and *LaTEX* for large specialty documents. After long deliberation, we have chosen Deneba's *Canvas* for technical illustration and Niles & Associates' *EndNote* for bibliography preparation. We have also been able to make good use of inexpensive file system repair utilities like Central Point's *MacTools*, Symantec's *SUM* and Alsoft's *Disk Express*. We bought a copy of *Mathematica* for the Mac II to tide users over until the faster free version becomes available for the NeXT machine.

We cooperated with some other laboratories on campus to purchase a site license for Coral's Allegro Common Lisp for the Macintosh. Although generally liked by the students who did small projects in this environment, this implementation was not employed in any big projects this year. To test Allegro Common Lisp's capability as a systems language we implemented a small utility that coalesces multiple lines of a TOPS-20 text file into unified paragraphs suitable for Microsoft Word's consumption. The overhead for using such a utility was relatively high — about the same as HyperCard's — but not unacceptable.

After we purchased the site licence (as did Carnegie Mellon and MIT), Apple Computer's Advanced Technology Group purchased the rights to Allegro Common Lisp and plans to continue to improve it and use it in their own research and distribute the improved version through APDA. We believe that this development will work to our advantage in the long term, making Apple Common Lisp a good, inexpensive vehicle for small AI applications.

We bought two copies of Connectix's *Virtual* software and Motorola PMMU hardware to experiment with virtual memory on the Macintosh. The product is mostly successful, but currently has problems with MacTCP and Allegro Common Lisp. Although basically successful, we have decided to defer purchase of more virtual memory hardware pending support by Apple and because of the cost.

MacTCP Stream Interface Package

Beginning last August, we alpha and beta-tested MacTCP — Apple's implementation of the lower levels of the Department of Defense's TCP/IP networking protocol suite. (The product manager for MacTCP is a former SUMEX employee now at Apple).

Before MacTCP, no TCP/IP-based applications for the Macintosh allowed simultaneous access to the network interface by other applications. Thus, one was limited to running a single network application at a time.

In writing Macintosh programs, accessing a driver is very complex and requires subtle calls to Mac OS primitives. For example, it requires 150 lines of code to open the MacTCP driver, create a low level I/O stream, and then open this I/O stream for reading and writing — and much of this code just defines “the bits” for a parameter block which is passed to the driver via a system queue. To eliminate these difficulties we wrote a high level stream interface to MacTCP — *tcpio*. *tcpio* permits one to easily access a remote host with five simple function calls: *tcpopen*, *tcplisten*, *tcpread*, *tcpwrite*, and *tcpclose*.

Tcpopen opens an active connection to a remote host, while *tcplisten* waits passively for connections from remote hosts. This latter allows one to run daemons in the background, e.g., a *finger* daemon which reports the status of the current user (if any) of a Macintosh. Each returns a TCP stream handle which is used in subsequent calls to *tcpio* functions.

Tcpread and *tcpwrite* read and write data on the open tcpstream, and *tcpclose* closes the connection. If one wishes to use asynchronous I/O—i.e., read and write without blocking—then three additional functions are provided: *tcplistencheck*, *tcpreadcheck*, and *tcpwritecheck*.

One calls *tcplistencheck* to see if a remote request for a passive connection has been made; *tcpreadcheck* to see if a read has been completed (and if so, how many bytes were read); and *tcpwritecheck* to check if a write has completed.

All of these functions return complete error messages in the case of an OS or network failure of some sort.

The marriage of *tcpio* and MacTCP provides excellent bandwidth on a local area network and will allow a single application to open up to sixty-four simultaneous TCP connections. One can read/write large blocks of data in excess of 400 kilobits/sec between a Macintosh and SUN-3 with both hosts on the Ethernet, and in excess of 150 kilobits/sec between the same two hosts when the Macintosh is on a LocalTalk network. Given that the bandwidth of LocalTalk is 230.4 kilobits/sec and that we use a Kinetics bridge between the LocalTalk and Ethernet, this performance is remarkable.

Domain Name Service

This spring MacTCP was in beta release and domain name service was added to the driver package. For reasons similar to those mentioned above, we wrote a high-level interface to this package. It allows one to do host name and address lookups, and uses a disk resident host file for familiar and frequently looked up names and addresses.

This package returns fully qualified domain names with each query, so, for example, if one looked up the name *SUMEX-AIM*, not only would one receive its network address—36.44.0.6—but also “SUMEX-AIM.Stanford.EDU,” in the reply. The converse is true for address probes. If a host has more than one address associated with its name, then all addresses are returned in the reply to a name lookup.

MacTCP - tcpio Applications

The above two packages are prerequisites for the writing of network applications on the Macintosh and are at the heart of MacMM, the mail reader/composer we are developing for this system.

An early first use of this package was to allow the Guardian project to distribute its analysis and results between Macintosh's and an Explorer. The Guardian BB1 program ran on the latter system, and sent data to Macintosh's for further analysis and graphics display.

We are also interested in doing remote database queries from the Macintosh to our SUN file servers. We have provided our *tcpio* package to Sybase so that they can provide TCP/IP connectivity from their Macintosh client to the Sybase database server that runs on SUN's.

Our *tcpio* package is in the public domain and will ultimately be distributed with MacTCP by Apple along with other public domain software that has been developed internally by Apple.

(6.2) Texas Instruments Explorers

The Texas Instruments Explorers have enjoyed an increasing popularity as more projects have developed a need for the combination of execution speed, full Common Lisp, and sophisticated development facilities offered by the

Explorer. The KSL use of Explorers has expanded to include 6 Explorer II's, 28 Explorer Is, and 16 microExplorers, for a total of 50 Explorer family systems. Our efforts have been directed at improving the environment of the Explorer by developing software, organizing user interest activities, and advising Texas Instruments.

Previous experience has shown that the greatest source of advancement for a particular computing environment is the user community. They are the most in touch with the deficiencies of the system, and thus uniquely positioned to address them, as well as to utilize the strengths of the system. The product developers of the system are frequently too involved in the lower levels of detail to produce general, effective solutions to problems, as well as being hampered by limited manpower resources. However, a significant amount of time and effort is required to organize this effort. This task has traditionally fallen to a user-run organization, such as DECUS or USENIX.

We have spearheaded an effort to organize an international users' group for the Explorer. The slightly misleading name for the group is NExUS, standing for National Explorer Users' Group. In the past year the NExUS steering committee has drafted a mission statement for the group, held a general meeting in conjunction with the AAI conference in St. Paul, moved towards a generally accessible library of user-contributed software, negotiated a relationship with TI-MIX, the general Texas Instruments users group, and maintained an electronic mailing list of several hundred people which has distributed over 400 messages among Explorer users world-wide.

We have maintained and extended the software tools produced in the KSL and made available to the national community. The tools now include:

- 36XX-EXPLORER
- BACKUP-TO-FILE-SYSTEM
- BATCH-PROCESSOR
- DEVELOPMENT-TOOL-CONSISTENCY-ENHANCEMENTS
- DVI-PREVIEWER
- EXPLORER-36XX
- GENERAL-INSPECTOR
- GRAPHER
- GRAPHICAL-VALUE-MONITORS
- IMAGEN-PRINTER-VIA-TCP
- INSPECTOR-ENHANCEMENTS
- KSL-PATCHES
- MAP-OVER-FILES
- MAP-OVER-FILES
- PATHNAME-EXTENSIONS

SEARCH-AND-REPLACE
SEARCH-AND-REPLACE
SINGLE-WINDOW-VT100
SOFT-KEYS
SOURCE-CODE-DEBUGGER
SPELLING-CHECKER
STRUCTURE-ENHANCEMENTS
UTILITIES
WEST-COAST-WINDOWS
WINDOW-ACCELERATORS
WINDOW-DEBUGGER-ENHANCEMENTS
WINDOW-ICONS
WINDOW-SYSTEM-ADDITIONS
ZMACS-ENHANCEMENTS

Many of the tools have been enhanced or newly written this year.

GENERAL-INSPECTOR The most significant of many improvements to this tool is the introduction of a "perspectives" mechanism, allowing the user to view a single data type (e.g. a list) as a representation for many different types of abstract data structures. For instance a list might be a mapping between tags and values implemented by a list of tag, value pairs (a Plist) or implemented by a list of sublists, the first element of each sublist being a tag and the second being the value (an Alist). Both example implementations have value in certain contexts, but often the user viewing such a data structure only wants to see what tags there are and what their values are. The perspective mechanism provides this. Also for example, a symbol might represent a named structure, a flavor, a CLOS class, a function, a type, a package, etc., and a perspective tailored to each of these is provided, as well as ways for users to conveniently add their own perspectives. This tool has also been extended to encompass the Common Lisp Object System and Portable Common Loops (using the perspectives mechanism).

WINDOW-DEBUGGER-ENHANCEMENTS This tool has been extended to use the General Inspector.

ZMACS-ENHANCEMENTS New features in this tool include: commands for formatting and pretty printing text and programs; commands for manipulating software systems defined with DEFSYSTEM; and more commands for dealing with Tag tables.

Of course, all of these will be provided to the user's library, and many of them have already been given to other sites, including IntelliCorp, Berkeley, ISI, University of Maryland, and Ohio State.

Third party software is less utilized, but we stay abreast of the latest releases of the expert system shell KEE. We have installed a DVI previewing system from MIT allowing TeX and LaTeX output to be viewed on the Explorer screen. We have available several other imported tools such as a Common Lisp LOOP package and we are experimenting with a Domain Name System Resolver.

In addition to producing and maintaining these software tools, we attempt to provide extensive testing and evaluation of Explorer hardware and software products in a sophisticated university research environment in order that these products work more effectively when they are distributed to the national community. This testing is critical to the development of the computing environment since the combination of concentrated in-house expertise and close links to the product developers allows a turn-around on problem fixes unavailable in the broader scope.

This year we have participated in testing TI's high-end product, the Explorer II-Plus, System Software Releases 4.0, 4.1, 4.1.1, 5.0, and 5.0.1, and the Common Lisp Object System. Many of these releases were focused on the relatively new microExplorer add-in co-processor for the Apple Macintosh II. Many of the suggestions we have made to TI in the course of using this machine have been integrated into the system. It is our hope that this work will result in high-performance, low-cost sophisticated Lisp availability, allowing greater dissemination of AI software to the national community.

TI's planned release date for their implementation of the emerging Common Lisp Object System (CLOS) standard has been moved forward by several months due in part to our urgings. Implementation of this standard will allow developers to write sophisticated, portable programs in an objected oriented framework.

In addition to specific testing and evaluation, we are constantly finding, tracking, fixing, and reporting software bugs. This year we submitted twenty-eight new bug reports on Explorer system software, almost all of which had fixes included. All of these fixes have been made available to the national community in a patch file.

As well as working on these specific problems, we have had many meetings with Texas Instruments representatives wherein we have attempted to present the needs of the national community for short- and long-term AI workstation products, covering issues including the desirability of specialized hardware, address space, programming environment versus execution speed, and the ability to utilize the AI workstation's power for routine tasks.

Of course, there is also a large number of day-to-day activities needed to keep the computing environment pleasant, including resource management (e.g., disk space allocation, printer management), assistance with file backup and magnetic tape usage, and introducing new users to the system. We have produced documents targeted at complete novice users, users of InterLisp-D machines, and users of Symbolics machines in order to facilitate user education. These documents have been used as examples at various places in the national community.

For the coming year we plan to continue development and maintenance of the software tools, perhaps adding facilities such as a DARPA Internet Domain Resolver, better IP access control, CLX and CLUE packages, better documentation, better software management facilities, a Zmacs novice mode, and an Explorer version of the TALK program, as well as aiding the growth of the users' group. We already have under development an Explorer IMAP mail client, UNIX LPD-based print spooling, and automatic file backup.

(6.3) SUN Workstations

Due to the high performance relative to purchase cost, SUN workstations have drawn strong interest as Lisp engines. For the past two years we have had three SUN 3/75 workstations in experimental use in the KSL. Because these were purchased for LISP work, we have added a 24 megabyte memory board (from Parity Systems Inc) to each of these. Also added were 70 megabyte SCSI disks. The systems have been set up to swap the large virtual memory to the local SCSI disk, rather than over the Ethernet to the server.

One of the systems is used for system development, a second has been used in the "Very Large Reusable Knowledge Base" project and a speech recognition box has been connected to another of the clients. An interface between the latter system and a Xerox InterLisp workstation running ONCOCIN was developed developed to study the use of speech input for medical information. An evaluation of SUN workstations was made in terms of their suitability as a platform for a general physician's workstation which would support data management, analysis and display, and consultative software. For now, the SUN/UNIX environment was judged not to be competitive, in terms of cost or user interface technology, with other workstations environments (e.g., the Mac II) for this purpose.

The vendor plans for LISP support on SUN workstations has been in a state of flux this past year. Despite the decision to stay with Lucid as a supplier of the underlying Lisp implementation, SUN has not made great progress

towards providing a programming environment competitive with older Lisp machine systems. However, SUN is continuing to work in this area and has the potential to close the gap, and thus deserves watching.

(6.4) NeXT Workstations

This year we obtained four NeXT workstations for evaluation and development, funded half by the SUMEX-AIM grant and half by other sponsored research sources in the KSL. We started following the NeXT closely in the middle of last year with several of the KSL planners attending pre-announcement, non-disclosure sessions about the machine. Several laboratory members attended the official, day long NeXT announcement and three of the SUMEX staff attended the following "NeXT Day" technical session. In early January, one of the staff attended the four-day NeXT Developer's Course and we received our first two machines. One of these machines was allocated for system integration purposes to a staff member and the other was made available for evaluation by laboratory members. A few weeks later our second pair of machines arrived, one of which was allocated to an Medical Computer Science application (OPAL) and the other installed in our Welch Road facility for use in the core AI research work of the Heuristic Programming Project. All machines were configured with 8 MB of memory and 330 MB SCSI Winchester disk drives, along with the standard NeXT 256 MB optical disk drive.

To accommodate the NeXT machines, we installed new thin Ethernet in parallel to our existing thick Ethernet in two of our locations. We then (with the assistance of technical notes from the AIR/SPUDS organization at Stanford) configured the networking software to be compatible with our environment. Rather than introduce four new file systems onto our network that would require backup and maintenance, we used the internal disks for the system software and swapping, but user directories were mounted via NFS (Network File System) from our existing SUN-3 file server, KNIFE. For authentication and host name resolution, we made the NeXT workstations "Yellow Pages" clients of our SUN server. This eliminated the need for separate account creation procedures for the new machines and made it possible for any user to use any of the NeXT's interchangeably. This did introduce some unique problems of designing user UNIX initialization files (.login and .cshrc) in such a way that they were both compatible with NeXT and SUN logins. Once the networking was in place we then got remote printing working using both our generic PostScript printers and our Apple LaserWriters, by means of our SUN-4 server as a spooling host.

To help users get started on the machine, we set up a local mailing list, "KSL-NeXT", to distribute information, which now has nearly fifty readers, several of whom are outside the KSL. We obtained access to the nation-wide NeXT-News mailing list and a campus-wide mailing list, NeXT-Info. We made available copies of every technical document that related to the machine, including "Objective-C" and "PostScript". We received permission for several of our programmers to audit a class conducted on campus by NeXT about

programming the machine. To further assist in learning to program the machines, we collected numerous non-trivial example programs with sources from public archives on the Internet and from the NeXT Developer's Course and made these available on-line.

Shortly after gaining some experience with the machines, we set up a meeting with Cindy Larson (sales representative) and David Bessemer (support engineer) from NeXT to discuss our status, progress, problems, and needs in using the NeXT machines. NeXT requested this meeting to learn about what we were doing with the machines and what was needed from them in the way of additional support and changes in future system releases. This meeting was quite successful and we have strongly influenced them regarding the need to fully integrate Common Lisp into their environment, particularly in the *Interface Builder* and *Application Kit*, and to make extensive use of the Common Lisp object standard in doing so. Also, we have sent in several dozen bug reports regarding the 0.8 release of their software. We are anticipating receiving the 0.9 system software release sometime this month.

We developed a pair of programs to control the display brightness and sound volume on the machines. These were both short-comings of the initial software release that we knew would be fixed in a later release but they were sufficiently annoying as to warrant some programming effort. These were well-received by the national NeXT community which was suffering with the same problems. Additionally, we developed a pair of conversion filters to facilitate moving sound files between the NeXT and the Macintosh. This allows us to use the built-in sound facilities of the NeXT to generate sound files for use in Macintosh applications, like *HyperCard* stacks. We wrote a bitmap conversion program to allow us to move bitmaps from our Xerox environment to the NeXT. Additionally, we are importing software from other universities, including both NeXT-specific applications as well as general Unix software such as *CSound* from MIT. We brought up the terminal-based, MM-like MS client for Unix, made available by Mark Crispin, which uses the IMAP protocol and software developed here. We anticipate Crispin making available a graphic mouse-based version built using the NeXT's Interface Builder, around the same time as we release our initial Macintosh version. We also installed a pre-release of the CMU portable speech library which included two speech recognition demonstration programs, *Sphinx* and *VSC* (voice spreadsheet). These are discussed in the Speech Project part of the report.

The OPAL project has made significant progress with the machine as the NeXT object-based interface construction kit fits naturally with that project's need to generate sophisticated interfaces programmatically. The machine has potential for use in our remote graphics and distributed computation work due to its Display PostScript server, which is accessible via Ethernet using TCP, and its Mach ports for remote procedure execution. This potential should start to become a reality with the next release of system software.

The machines have had an unexpected positive impact on the Speech project due to both their ability to run the CMU speech software as well as provide faster cycles than the SUN 3 currently being used to process speech using the SSI software library.

We had two early hardware failures, one of which was minor and both of which were repaired under warranty; all four machines are no longer under warranty. We have no plans to obtain more NeXT machines but their numbers continue to increase campus-wide and in the AIM community.

(6.5) Xerox D-Machines

This year saw the introduction of a new release of the Xerox Lisp environment, a new hardware platform, a new release of server software, and a new company. The use of Xerox workstations in the KSL continued to decrease though it has not yet disappeared completely. Most of the SUMEX-AIM systems software projects on these machines consisted of aids to transfer to other platforms and small prototype systems. As part of our removal of support for the equipment, we turned over to Ohio State University responsibility for the national INFO-1100 Xerox Lisp interest Internet mailing list which we have been operating since the machines first appeared seven years ago. We also discontinued the Internet software repository for Lisp programs on SUMEX-AIM, in conjunction with the SUN 4 transition, and this was picked up jointly by Xerox and Ohio State.

As mentioned very briefly in the previous annual report, we were involved early in the evaluation and testing of a new hardware platform, known as Maiko, which consisted of a Xerox Lisp byte code emulator, implemented on a SUN workstation in 'C' with some optimization of the machine code. The byte code emulator provided binary compatibility with the Xerox hardware thus making porting of software trivial. We used the ONCOCIN system to evaluate the new platform. Using the full range of SUN 3 and SUN 4 processors we saw performance ranging from half of the Xerox 1186 to several times faster than the Xerox 1132. Although technically quite impressive, in general we did not consider this as a means to prolong our use of the Xerox Lisp environment, due to the cost of the SUN hardware necessary to get the performance we would require. Maiko was seriously evaluated as a possible short term fix for some performance and packaging problems in both the clinic ONCOCIN system and the speech project, though later rejected. Xerox also saw this as a short term solution to the problem of moving onto more conventional hardware.

To facilitate the new platform, Xerox provided a new software release, Medley, for which we were involved in two separate beta-tests. We were one of two sites involved in the initial beta-test and this consisted of the usual testing and fixing of existing software and reporting of problems. For the most part, the new release was a better version of their previous combined InterLisp and Common Lisp release, Lyric, with just a few feature additions. This release provided increased support and faster throughput for TCP/IP-

based network communications in both the Lisp and SystemTool (Lisp installation) environments. It also provided several UNIX compatibility improvements. As part of the new release, we beta-tested other revised software modules, such as Xerox ROOMS, and updated all of our Lispusers modules for re-release as well as added some new ones described further on. Since this release occurred around the time our user community moved onto the TCP/IP-only SUN 4, we incorporated the TCP networking ability into our full lisp system, instead of a separate loadup and later made this the case as well with the previous release, Lyric. We also introduced a PostScript driver into our full loadup at the same time in accord with our changing printer situation. Since Medley proved to be more stable than Lyric, we shortly dropped support for Lyric and continue to use Medley along with the InterLisp-only Koto environment for performance reasons.

The new Medley/Maiko system was introduced at AAI (for which we provided a demonstration application) along with a spin off company, ENVOS, that Xerox launched to develop and market their AI workstation products. ENVOS, by being a smaller, more aggressive company, was intended to solve a number of problems that Xerox AI systems division had experienced marketing their products. Though there was much early excitement in the technical press over this new company, late last year they were laying off employees and by early in the second quarter of this year the company was out of business. Since we had made a decision early on not to continue to support use of this environment in our laboratory, these events had little more than passing interest to us as we were not actual customers of ENVOS, except for a Xerox University Grant program (UGP) connection very near the end. The commercial future of Xerox Lisp remains unresolved.

Although in the previous annual report we were anticipating going off the extended Xerox maintenance agreement provided by the UGP, this maintenance agreement was extended for one more, final year. The agreement, which covers the Xerox file, print and boot servers as well as 15 1108/9 Lisp processors, will end shortly, as of May 31st of this year. Based on our current reduced usage and funding constraints we will not be extending this contract further. All of the Xerox servers were upgraded to the new Services 11.0 over the course of the year. This led to a problem where older releases of Xerox Lisp could no longer access the printer. After tracking down the problem, a fix was quickly provided by the vendor. Additionally, the Xerox boot service we had been running which is vital to the installation and maintenance of the equipment was not provided under the new release of services. We are still trying to resolve this issue with Xerox, before our maintenance agreement expires. In the mean time, we have been running our Lisp-based boot server which we implemented several years ago before Xerox made this service available on their servers. Unfortunately, this server is significantly slower than the Xerox Mesa-based one and our maintenance operations on the Xerox equipment have been much less efficient.

We continue to operate half of our 1186s (13 out of 24), nearly all of our 1109's (11 out of 15) and none of our 1108's (15). We obtained a donation of six 1108/9's that were used for parts to repair broken machines and upgrade others from 1108's to 1109's. We are currently supplying two 1186 machines to projects outside the KSL and have provided three 1109's to a Molecular Biology project in which several MIS students participate. We are using one 1186 as a network monitor, running the Xerox Mesa development environment, at our Welch Road facility. We are also still using our Xerox 1132, though it is scheduled to be removed from use shortly when we remove an obsolete 3 MB experimental Ethernet. In order to facilitate the video recording of demonstrations of projects that have been completed on the Xerox equipment, partially in anticipation that such equipment will not be available for live demonstrations in the future, we retuned an 1108/9 display to be compatible with the video tape scan rate.

Although the 1186 is a more recent product than the 1109 we still use 1109's as they are equally powerful, more reliable and were, until recently, under service contract. The 1186 has proved somewhat less reliable though we continue to use as many as are fully functional and tend to use them in either locations that are further removed from our main building as they are significantly easier to transport or for applications requiring larger screens. We removed from service 15 1108's that were not part of the UGP, replacing them with either unused UGP 1109's or 1186s. The 15 unused 1108's were then collected together, tested and repaired in anticipation of selling them (at less than 5% original cost) back to Xerox. Unfortunately, despite the effort in preparing the machines, this arrangement fell through and we are still in the, potentially fruitless, process of finding a buyer.

As part of our transition away from the Xerox Lisp environment, we developed a tool to allow graphics in various forms to be transferred to the Apple Macintosh with minimal loss of information and maximum flexibility. It was important to both provide a way to transfer drawings as well as provide a way for users writing new documents using the Macintosh to include illustrations derived from work done using the older system (e.g. the ONCOCIN technical specification documents). We initially experimented with, and put some programming effort into, a utility provided by another University which did conversions between Xerox bitmaps and MacPaint documents. This utility basically worked but was limited to MacPaint's relatively small image limit (576 by 720 pixels) and could only handle bitmaps, not higher level graphic representations.

To overcome this first obstacle, limited bitmap size, we wrote a new utility that could convert Xerox bitmaps into Macintosh PICT bitmap format, one operator in the PICT graphics representation. Using this, we were able to transfer bitmaps larger than the screens of any of the systems we currently use. Once the files were converted and transferred to the Macintosh, some minimal fix-up was then needed to set the proper file type before the file could be read into one of several commercial and shareware applications.

This utility was later incorporated as a small piece of a much more ambitious program that addressed the second problem, high level graphic information transfer.

To convert most types of graphic data from the Xerox Lisp environment to the Macintosh required the ability to do it in such a way that the data was still manipulatable as abstract graphic entities (lines, curves, text, bit images, etc.). The PICT representation included these graphic abstractions, was reasonable, and sufficient for our needs. The biggest challenges to the technical and visual success of such a conversion were the problems related to the manipulation and mathematics of moving between the Xerox and Macintosh font representations, so this problem was attacked first.

With the intention of building a general purpose graphics conversion program, we wrote a utility that allowed the Xerox generic graphics driver to read Macintosh font files and do font metric calculations. As a side benefit the utility also extended the number of display fonts available in the Xerox Lisp environment. Once completed, this program was made available separately to the national Xerox Lisp community as READAPPLEFONT, part of the Medley release Lispusers modules. Once the problem of font manipulation was in hand, work began, sometime later, on the full conversion program.

The next step involved implementing a full graphics driver for the PICT format based on our earlier graphics drivers (Impress, HPGL, CDI, etc.). This driver appeared to the user on the Xerox Lisp machine as printer file type (similar to PostScript, Interpress, etc.) but in fact operated at a higher level than any of our previous graphics drivers. This driver preserved grouping information so that dashed lines, polygons, etc. did not get broken into individual lines as usually happens when a document is converted into a laser printer master. The graphics driver took more time to debug, fine tune and polish than earlier ones as it incorporated much more detail (including color and 16-bit to 8-bit character set conversions) and the Macintosh proved relatively hostile and uninformative when it came to debugging PICT data files. The READAPPLEFONT module was also further refined to handle subtle issues regarding bold and italic font widths.

Once operational, the PICT conversion program was fine tuned by transferring existing graphics documents obtained from as wide a range of sources as possible, evaluating the results and modifying the behavior of the program accordingly. A large collection of Macintosh font files were extracted into Xerox Lisp readable files to facilitate this. Once converted to PICT representation, the Xerox graphics were able to be read and, most importantly, manipulated using program such as MacDraw I & II, Canvas, GrayView, Giffer, etc. This process also gave us another means to evaluate the various graphics drawing programs available for the Macintosh in our search for a reasonable one for laboratory-wide use. The completed program was used to convert numerous documents, bitmaps and other graphics entities and continues to be used on a regular basis.

Some student time to do conversions was provided for users who had documents that need to be transferred but who have distanced themselves from the Xerox Lisp environment. To further assist users, even though the program was developed under the Medley release of Xerox Lisp, it was patched and made available under the earlier Lyric release which was still in use at the time. The conversion program was released early in this year to the national Xerox Lisp community as the PICT Lispusers module.

A project was undertaken by a student programmer to implement a set of Common Lisp routines to index and interrogate a dictionary database. This project was done in Xerox Common Lisp and later tested under Allegro Common Lisp on the Macintosh. This package was then incorporated into a simple, experimental server that conformed to the TCP/IP Finger protocol. The host on which the server ran, a dedicated Xerox 1109, was aliased with the name Webster and users were then able to do a dictionary lookup from any of our hosts by doing: *finger word@webster*. This provided a simple network-wide dictionary facility without the overhead of storing the large dictionary database on each host that needed nor the effort of implementing new client software for each type of host. For debugging purposes, a Finger client implementation was done for the Xerox Lisp environment as well. The server continues to be used on an experimental basis.

Another project involved building a remote screen facility to allow one Xerox workstation to open a window onto another via the Ethernet to provide remote access and control of the screen, keyboard and mouse. Intended for evaluation purposes only and not for actual use on the Xerox workstations, the program was developed there using the superior networking and prototyping facilities. This program is described in more detail in the remote/distributed graphics portion of this report.

(6.6) Symbolics Lisp Machines

We have terminated all support of Symbolics workstations. As has been stated previously, in order for workstations to be competitive with time-shared mainframe computing resources, they must not only have a low purchase price, but must be cost-effective to maintain. This goal is normally achieved due to the economies of scale associated with having a large number of identical parts in an installation, as well as amortizing the cost of software development over many machines. We have come to reasonable agreements with all of the workstation vendors except for Symbolics. The high costs of service, the exceptionally high price of mail-in board repair, and the lack of a reasonable self-service alternative for the L-series workstations (3600, 3640, and 3670) has left us unable to justify continued support of these machines.

We are reluctant to get involved with one of Symbolics' newer products for three reasons. First, our experience with the L-series machines makes us wonder if Symbolics will drop support for current equipment as rapidly in the future. Second, we are leery of the occasional indications that Symbolics might not be around for much longer. Finally, there is little to recommend

Symbolics' products over their competitor's. However, Symbolics is still a player, and we will track their developments accordingly.

(6.7) HP 9836 Workstations

Three of the four HP 9836 workstations, which had been taken out of use, were provided to Dr. Craig Miller's group in Cardiovascular Surgery here at Stanford on indefinite loan. Dr. Miller's group already had similar equipment and was in need of the extra CPU cycles and keyboards our unused equipment could provide. We shut off the remaining HP 9836 after the arrival of the NeXT workstations as the HP/UX workstation was no longer needed for UNIX cycles. We retained the external Conrac color displays which were part of the HP equipment donation and retuned them to be compatible with our Macintosh workstations for use when a multiple display presentation or color is required.

(7) Remote Workstation Access, Virtual Graphics, and Windows

(7.1) Remote Access

The move towards a distributed workstation computing environment for AI research in the SUMEX-AIM community means that a number of technical obstacles must be overcome. For example, in order to allow users to work on workstations over networks from any location — at work, at home, or across the country — we must facilitate connecting a user display remotely with a workstation computing engine. The first step has been making reliable terminal access operational on all workstations, based uniformly on TCP-IP protocol services. This allows primitive (non-graphical) access between workstations. A more comprehensive access can be provided through remote virtual graphics connections.

(7.2) Virtual Graphics and Windows

In order to link the output of workstation displays across networks, it is necessary to capture and encode the many graphics operations involved so that they can be sent over a relatively low-speed network connection with the same interactive facility as if one had the display connected through the dedicated high-speed (30 Mhz) native vendor display/workstation connection. A mechanism for doing this is called a *remote graphics protocol*.

The *X window* system¹, developed under the MIT Project Athena, has become a widely touted remote graphics protocol standard. X is a very complete protocol that operates in a client/server fashion, where *server* in X terminology refers to the program running on the user's display and *client* refers to the program running on the remote host/workstation. Despite its wide publicity, X is not being adopted as the *core* window/graphics model by

¹ Scheifler, R. W., and Gettys, J. "The X Window System." *ACM Trans. on Graphics*, , 1986.

many of the major workstation vendors. For example, Apple continues to use a proprietary protocol, NeXT has adopted Display PostScript as its imaging model, and SUN is still ambivalent between NeWS and X. Nevertheless, working X Windows implementations are available now for SUNs, MicroVAX's, the DEC MIPS workstation series, and soon will be available on Macintosh's from a third party vendor. Common Lisp X client and server implementations also have now been released for TI Explorers (CLX) and include a supporting CLX library for the creation of windows, menus, scroll-bars and other graphical objects. Further, The Common Lisp User Environment (CLUE) is also available from TI and is a higher-level window system on top of the primitive CLX library which uses the Common Lisp Object System (CLOS). Since CLUE is a more general window-system/user-environment, it satisfies the long standing need for a portable window system for developing Common Lisp AI applications and will be applied here at SUMEX-AIM, for example in the porting of the ONCOCIN system from InterLisp to Common Lisp.

Because of the severe funding pressures this year and the need to devote almost all of our resources to the transition from the DEC 2060 to the SUN-4 computing environment, we have not been able to play a very active role in developing remote graphics access and related general system and user tools to our computing environment. Our early work on integrating the Stanford University V window system, and its virtual graphics terminal protocol based on structured display files, in the operating system of a Xerox 1186 and on writing an initial X client for the Explorer using the alpha release of the X.11 specification layed a good foundation. While diverted to our other tasks, workstation vendors have provided us with a more extensive and highly integrated set of tools. Our emphasis in the area of remote access/graphics has evolved from proving the concept of remote windows and remote tools to building real systems on top of remote access capabilities for routine use.

(7.3) Remote Graphics Applications

DISPLAYWATCH

We made a number of experiments with the *TIMBUKTU* remote machine manipulation program for the Macintosh which allows control via the network of the keyboard and mouse as well as remote viewing of the display. When used with a small screen Macintosh as the remote host over combinations of *AppleNet* and Ethernet segments, the program was somewhat sluggish. However, when tried between two machines with Ethernet boards, the response was quite good (as good as a smaller Macintosh when used directly). We also tested it between a staff member's home Macintosh and a Macintosh at Stanford via the *Shiva NetSerial* gateway. This worked, to the *NetSerial's* credit, but was far too slow when using a 2400 baud modem.

There were some problems with the software and a few annoyances that would make using it routinely a problem, particularly the unique serial

numbered copies aspect — understandable, but it makes building servers on other types of machine much more difficult. Although it worked with our 19 inch and smaller displays, it did not work with our 24 inch displays despite the fact was supposed to handle any size display. To further experiment with the style of graphics protocol employed by this software, we built an equivalent of the program using the Xerox Lisp environment. This program was not compatible with the Macintosh one, but rather was optimized for the Xerox workstation as the program needed to be as fast and efficient as possible.

This system, *DISPLAYWATCH*, allowed a user to open a window on one Xerox workstation that would become the screen for another via the Ethernet and facilitate control of the remote keyboard and mouse using the local ones. Within the window, everything worked as if you were on the remote machine. The goal was to put together a working remote display system, by whatever means, to allow experimentation. The system took the approach that remoting everything on the Xerox workstation screen using a high level graphics protocol was impossible due to the fact that too many programs manipulated the screen bitmap directly, this was also true of the Macintosh. So, instead the program used a very low level approach that looked for bit (word) changes on the screen and sent update information across the network.

The system was layered on a reliable byte stream and worked using either the *SPP (XNS)* or *TCP/IP* protocols, though other types of byte streams could also be added (like *RS-232*). The system was nearly 100% functional, as compared to the local display, keyboard and mouse, and was reasonably stable. It included such details as interrupt character processing and control over remote and local mouse clicks. The only real problem was that when used between 1186 and 1109 class machines it was very slow. This was due to the CPU intensive server trying to find changes on the remote screen bitmap, not a communications problem. We further tested it using the faster Xerox 1132, in hope of potentially using this program as a solution to increasing the speed of the ONCOCIN clinic system without introducing a larger piece of hardware into the clinic. Unfortunately, although the improvement in response was quite encouraging, the program was not fast enough to solve the clinic problem.

This program was later demonstrated to several systems people from Xerox and ENVOS in hope of their taking on the task of further development and integrating it at a much lower level into their environment to provide the necessary speed. Unfortunately, though interested, they lacked the resources to take on such a project and development on the program has concluded. What the program was able to achieve, however, has helped influence our thinking regarding both the applications of remote graphics as well as the variety of implementations possible.

TALK

The TALK system (intra-machine user communication tool which employs both text and graphics) was substantially reorganized and had several improvements added. The previous release of the TALK program was contained in a single module that would detect what was contained in the environment (*TEdit*, *TCP*, etc.) and make services available accordingly. The new TALK was completely broken out into individual modules (three services, three network protocols and the main program) which could be loaded independently based on need. This fixed a number of problems and only introduced a much smaller set of others. Several bugs related to the new release of the Xerox Lisp environment, *Medley*, were also tracked down and solved (some by the vendor).

As part of convincing ourselves that the program was layered correctly, we added a *PUP* protocol module. We were able to get this up and running quite quickly without any modification to the TALK program itself. This protocol used a simple *PUP* packet exchange protocol to negotiate the connection and then switched to using a *BSP* byte stream. This was only meant to test the design of the TALK program and the additional protocol was not released to the Xerox Lispusers community. TALK had always determined which service (*TTY*, *TEdit* or *Sketch*) to use based on what the two communicating machines had in common. However, previously, it would map over the known network protocols (*XNS* and *TCP/IP*) and just use the first one that was able to resolve the destination name into an address, and if there was no server on the other side, it would then quit. It was improved to be able to try all the protocols (*XNS*, *TCP/IP* and *PUP*) that resolve the destination name in turn to see if it could locate a server with any of them.

Despite the reorganization, TALK remained identical at the network interface so the revised TALK was completely compatible with older versions. The user interface was kept nearly identical as well. The Xerox Lisp implementation work on TALK has been concluded and the revised version was provided to the national Xerox *Lispusers* community as part of the *Medley* release. There are no plans to continue work on this implementation.

Other Remote Tools

Last year we finished a tool (MONITOR) that allows a Xerox Lisp machine to examine the display of another workstation. This tool shows a shrunken version of the entire remote machine's display in one window and allows you to examine a smaller portion of the display in full size in another. We still plan to use this to remotely examine the display on the Oncology clinic machine (which runs the ONCOCIN expert system) when one of the physicians calls up with a problem. This tool is another built on top of the Courier server we described in previous years.

(8) Network Services

An important aspect of the SUMEX system is effective communication within our growing distributed computing environment and with remote users. In addition to the economic arguments for terminal access, networking offers other advantages for shared computing. These include improved inter-user communications, more effective software sharing, uniform user access to multiple machines and special purpose resources, convenient file transfers, more effective backup, and co-processing between remote machines. Networks are crucial for maintaining the collaborative scientific and software contacts within the SUMEX-AIM community.

(8.1) National and Wide-Area Networks

ARPANET Status

Since the early 1970's, the ARPANET has been the primary link between SUMEX and other university and AIM machine resources, including the large AI computer science community supported by DARPA. It provided key tools for collaboration and software sharing, including electronic mail transfer, remote terminal connections, and other file transfers.

Major changes have been underway over the past year in the allocation and management of ARPANET resources. DARPA has sought to limit the funding it allocates to network operations in order to focus more of its efforts on networking research and long-term developments. It does not want the mandate of organizing or running a broad national research Internet. In the spring of last year, DARPA began the reconfiguration of the ARPANET to eliminate all connections other than those directly related to DoD applications. An implication of these changes has been the elimination of almost all university ARPANET connections and their replacement by NSFNet links. Early in April of 1989, the connection to our DECSys-20 was deactivated. Although ARPANET equipment is still on-site to continue the basic trunk-line connectivity of the network, it will soon be removed after ARPA contractors finish reconfiguring the network in the bay area to a smaller scale.

In the longer term, DARPA's role will be limited to DoD communications research and support of its contractor community. Thus, the ARPANET has once again become very restrictive in terms of access and use. DARPA expects to construct a Defense Research Internet (DRI) over the next several years (in cooperation with DCA and SDI), with the current ARPANET completely disappearing after that.

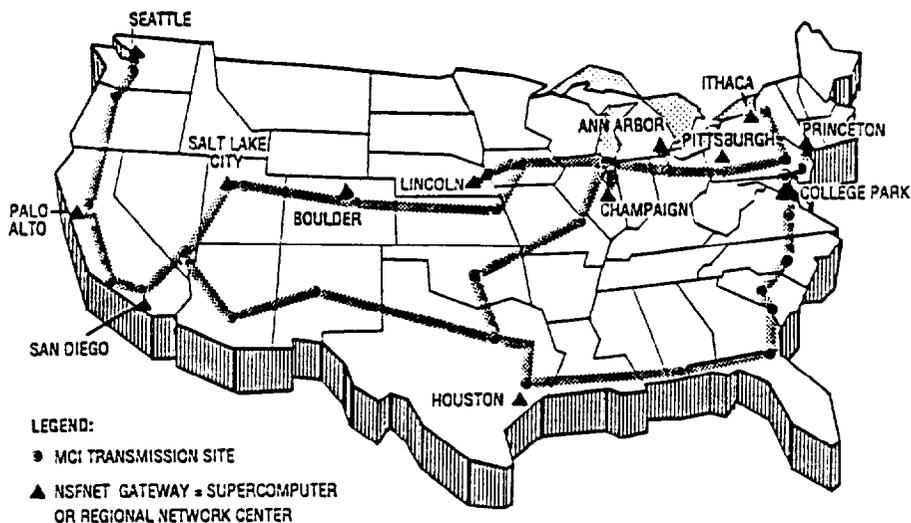


Figure 1. NSFNet Configuration as of January 1989 (from Marshal)

NSFNet Status

The NSFNet has been broadly announced as a national research network¹ and consists of a national backbone network coupled to a number of regional networks that link research institutions with each other and with the backbone (see Figure 1). The backbone operates with T1 (1.5 Mbit/sec) links between nodes. The current network is being managed by MCI, Inc. with MERIT Corp. (a nonprofit consortium of 8 Michigan universities) and IBM as research partners. The backbone links the 6 current NSF supercomputer sites and 7 regional academic computing networks.

The Bay Area Regional Research Network (BARRNet), to which the Stanford University Network (SUNet) is connected, now provides the primary link between SUMEX-AIM and the national Internet community. The upgrade of the backbone services to T1 links has improved network throughput and responsiveness significantly over the previously highly-congested 56 Kbit/sec ARPANET services. The NSFNet/BARRNet services are currently funded by NSF and Stanford University so that NIH derives the benefits without any direct investment, as was the case with the earlier SUMEX-AIM connection to the ARPANET which was funded by DARPA as part of its support of KSL basic research activities.

¹ Marshal, E. "NSF Opens High-Speed Computer Network." *Science*. 243: 22-23, 1989.

TELENET Service

SUMEX continues to be served by the commercial network, TELENET Inc. As reported previously the SUMEX connection to the TELENET is by means of a four-way gateway (developed by Develcon, Inc. — see Figure 7), which provides protocol translation between the X.25 system used by TELENET and our TCP/IP-based Ethernet system. Though somewhat experimental when set up, it has provided a reliable connection to TELENET. Though it is primarily for users coming into the SUMEX system, it also provides outbound connections. Thus users can be at their normal SUMEX terminal and access hosts on the TELENET system.

The Stanford Lane Medical Library catalog and circulation system (LOIS), which runs on a Tandem computer system, is connected to the Develcon gateway as is a leased line to the bibliographic search facilities of Dialog Inc. These links, under the direction of the Lane Library, provide SUMEX users with access both to the database of Lane Library materials and to the MEDLINE service provided by Dialog. The latter is a replacement for a similar service provided by BRS Inc. which was terminated in April of this year. As with the LOIS system described above, SUMEX users can also access the MEDLINE service much like they would other local services on the Ethernet.

In spite of the potential for greatly improved functionality and information resource access, the style of the interface between X.25-based nets and TCP/IP-based nets is not very smooth. Vastly different approaches to addressing, connection concepts, and terminal handling, make a general solution to the interface problem very difficult.

(8.2) Local Area Networks - LAN's

In the past, we have developed local area networking systems to enhance the facilities available to researchers. Much of this work has centered on the effective integration of distributed computing resources in the form of mainframes, workstations, and servers. Network gateways and terminal interface processors (TIP's) were developed and extended to link our environment together. A diagram of our local area network system is shown in Figure 8 and the following summarizes our LAN-related development work.

Workstation Networks

As noted in last year's report, most of our Apple Macintosh computers are connected with *PhoneNet* products. They are integrated with the rest of our equipment by connecting the PhoneNet networks to the campus Ethernet networks using *Kinetics FastPath* gateways, a commercial spin off resulting from the SUMEX work on the SEAGATE gateway. Kinetics licensed this software/hardware from Stanford University and used it as a basis for its product.

This year we added several Kinetics AppleNet-to-Ethernet gateways to support our Mac II's, LaserWriters, and MicroExplorers to link them with the Stanford Ethernet system and thereby to the NSFNet Internet. Kinetics introduced a new model gateway — the *FastPath-4* — which incorporated a faster processor, more memory, and new software. We delayed a portion of our Macintosh network installation last year so we could take advantage of this new model when it finally became available. Our benchmarks demonstrated that the new model improved throughput by factors of 2-10. By using Kinetics purchase credits, we may be able to replace our remaining model 1 and 2 FastPath gateways with model 4's this year.

AppleShare protocol file service for the Macintoshes is provided by two SUN file servers and a VAX file server running the CAP AppleShare file server software.

We also purchased a *Shiva NetSerial X-232* AppleTalk Dial-In server to experiment with dial-in network access. This device is discussed elsewhere in this report in the section titled "Distributed Information Resources and Access".

Ethernet Gateways

In our heterogeneous network environment, in order to provide workstation access to file servers, mail servers, and other computers within the Stanford local area network, it is necessary to be able to route multiple networking protocols through the network gateways. As reported last year, the SUMEX gateways support PUP, Xerox NS, Symbolics/Texas-Instrument CHAOSNet, and the TCP/IP protocols. This support not only provides the routers necessary to move such packets among the subnetworks, but also other miscellaneous services such as time, name/address lookup, host statistics, bootstrap support, address resolution, and routing table broadcast and query information.

This year we began the transition away from the SUMEX developed gateways and TIPs to the Cisco Systems, Inc., gateway software and hardware. Cisco Systems' software products are derived from the SUMEX software as licensed through the Stanford University Office of Technology Licensing. The Cisco gateway servers also support the diverse protocols we use and are compatible with our environment. We were given Cisco processors by the Stanford IR-Networking group, as well as non-volatile RAM memory boards (NVRAMs). These systems have battery backup and contain configuration information essential for each gateway. Once the NVRAMs are initialized, the gateways can load their software from on-board PROM's and their configuration information from the NVRAMs after a power failure so that remote booting of these critical network nodes is no longer necessary. We believe this step will free our staff from more routine maintenance tasks so that they can concentrate their time on development.

Terminal Interface Processors

SUMEX-AIM has five TIPs, and in previous years we spent a significant amount of time maintaining and augmenting this software. Over the past year, as with our Ethernet gateways, we migrated the EtherTIP software to the Cisco Systems terminal processor software. This software is available under site license to Stanford University. It has many additional features such as Serial Line IP (SLIP) support, serial port dial out, and the NVRAM power failure protection feature mentioned above.

Currently, one of our TIPs has ten dial-in ports and is used extensively by the local SUMEX-AIM community for dial-in access to the SUMEX-AIM SUN-4, SUN-3, and VAX 11/750 servers from home during off hours. The four remaining TIPs are used to access various mainframes during work hours.

(9) Distributed Information Resources and Access

As mentioned in last year's report, there are many user needs for getting information from and about the computing environment, ranging from help with command syntax to sophisticated database queries. A distributed computing environment adds new complexities in making such information accessible and also new requirements for information about the distributed environment itself. We began to adapt the many workstation-specific information tools to include distributed environment information such as workstation and server availability, "Finger" information about user locations and system loads, network connectivity, and other information of interest to users in designing approaches to carrying out their research tasks. In addition, this year we wanted to develop general systems tools for monitoring and debugging distributed system performance to identify workstation and network problems. Finally, we still need to adapt and develop distributed system tools for remote database queries and organize the diverse sources of information of interest to AIM community members to facilitate remote workstation access to community, project, and personal information that has traditionally existed in ad hoc files on mainframe systems.

Only marginal progress has been made toward these goals this past year because NIH budget cuts forced us to place most of our efforts in the early part of this reporting period on shutting down the 2060 and moving the SUMEX-AIM users to the SUN-4. This move has been completed, but as a result, we have not had as much time to devote ourselves to the research needs required to support our distributed environment as we had hoped to have during this year's reporting period.

Still, the Macintosh HyperCard tool provides a very powerful environment in which to hierarchically organize and provide access to information in the form of text, graphics, pictures, and even sound. This year we began the development of a "KSL stack" for HyperCard that will eventually include descriptions of KSL and AIM personnel, SUMEX-AIM projects, SUMEX-AIM computing systems, KSL building maps, the KSL bibliography, etc. In

response to a user request to share data in a HyperCard stack among several workstations, we investigated and developed a work-around that let them simultaneously access the stack on a SUN file server. When Apple releases HyperCard version 1.2.2 this work-around will no longer be necessary.

On another front, as we reported last year, in conjunction with the SUN file server we had mounted an experimental database system for remote information access using the commercial UNIFY database product. Our goal was to make access to the database information possible from a distributed workstation environment through network query transactions, as opposed to asking the user to log into the database system as a separate job and type in queries directly. This will facilitate remote information access from within *programs* including expert systems, where the information can be filtered, integrated with other information, and presented to the user. Such a system will provide multi-user, multi-database access capability; that is, several users will be able to have access to a single database at the same time, and a single user will be able to have access to several databases at the same time.

Since last year we have been investigating other database products that would more fully integrate our Macintoshes as well as Explorers and Micro-Explorers with databases resident on SUN file servers. There are other commercial databases besides UNIFY available, e. g., SYBASE, and front ends like CL/1 that will allow one to use slightly divergent client implementations of SQL in such a way that CL/1 will post the query to the appropriate database server, be it UNIFY or SYBASE. These clients still lack adequate TCP/IP support, and we in fact, have offered our TCP/IP stream package to these vendors to encourage its use in their products.

Also, as an experiment, we developed an English-language dictionary server on a Xerox 1108. Any host implementing the IP protocol *Finger* client can access the server without additional programming. (See the *Xerox Lisp Machines* section of this report for further details.)

As an experiment in remote network access for Macintosh workstations, we purchased a *Shiva NetSerial X-232* AppleTalk Dial-In server to give us dial-in network access. A researcher (or collaborator) with a Macintosh at another location (perhaps his home) can use the NetSerial software and a modem to connect his Macintosh (via the telephone connection and the NetSerial device) to one of the Phonenet AppleTalk networks at SUMEX. Then the researcher has access to SUMEX's file servers and printers as though his Macintosh were connected directly to the network. Owing to the low bandwidth of telephone lines, it has turned out that throughput discourages all but the most patient user. We experimented with TCP connections from a Macintosh at a private home to non-AppleTalk hosts at Stanford via the NetSerial. Although this experimental configuration worked surprisingly well, we plan to experiment with the SLIP protocol in the coming year. The SLIP protocol is potentially more efficient and is more widely implemented.

Finally, with a new version of the MacTCP-based *SU MacIP* TELNET program (which we are currently alpha testing) our users will be able to

initiate dial-out terminal sessions from their Macintoshes to remote hosts via the networks and EtherTIP, which has 10 telephone lines and modems attached to it. The EtherTIP now has software to support this type of dial-out service. Using the current version of SU MacIP, users are already able to access the MEDLINE bibliography service provided by Dialog via our networks and the Develcon gateway.

(10) Distributed system operation and management

As mentioned in last year's report, the primary requirements in this area are user accounting (including authorization and billing), data backup, resource allocations (including disk space, console time, printing access, CPU time, etc.), and maintenance of community data bases about users and projects. Our accounting needs are a function of system reporting and cost recovery requirements. The distributed environment presents additional problems for tracking resource usage and will require developing protocols for recording various kinds of usage in central data base logs and programs for analyzing and extracting appropriate reports and billing information. We are still involved in analyzing the kinds of resource usage that can be reasonably accounted for in a distributed environment (e.g., printing, file storage, network usage, console time, processor usage, server access), and investigating what facilities vendors have provided for keeping such accounts. Data backup is, of course, closely related to the filing issue. We continue to use and improve network based file backup for many of our file servers.

III.A.2.5. Relevant Core Research Publications

The following is a list of new publications and reports that have come out of our core research and development efforts over the past year. In addition, we include references to earlier reports that are discussed in the "Progress Summary" text above.

- KSL 87-22 Homer L. Chin and Gregory F. Cooper; **Stochastic Simulation of Casual Bayesian Models**, November 1988. To appear as a chapter: *Bayesian Belief Network Inference Using Simulation*, in the forthcoming book: *Uncertainty in Artificial Intelligence 3*, L. N. Kanal, T.S. Levitt, and J.F. Lemmer, eds., North-Holland. 20 pages.
- KSL 87-50 (Journal Memo) Mark E. Frisse; **Searching for Information in a Hypertext Medical Handbook: The Washington University Dynamic Medical Handbook Project**, August 1987. 7 pages Has appeared in: *Communications of the ACM*, July 1988, Volume 31, Number 7, pp. 880-886.
- KSL 87-70 Tu, S.W., Kahn, M. G., Musen, M.A., Ferguson, J.C., Shortliffe, E.H., and Fagan, L.M.; **"Episodic Monitoring of Time-Oriented Data for Heuristic Skeletal-Plan Refinement."** August 1988, April 1989. 45 pages.
- KSL 88-18 Peter D. Karp; **A Process-Oriented Model of Bacterial Gene Regulation**, June 1988. 14 pages
- KSL 88-29 Andrew Gelman, Susan Altman, Matt Pallakoff, Ketan Doshi, Catherine Manago, Thomas C. Rindfleisch, and Bruce G. Buchanan; **FRM: An Intelligent Assistant for Financial Resource Management**, April 1988. 25 pages. *Proceedings of The Seventh National Conference on Artificial Intelligence*, pages 31-36, August 1988.
- KSL 88-33 Bruce A. Delagi and Nakul P. Saraiya; **Elint in Lamina, Application of a Concurrent Object Language**, presented to: *Workshop on Object-Based Concurrent Programming-OOPSLA 88.*, July 1988, 13 pages.
- KSL 88-35 Eric J. Horvitz; **Reasoning Under Varying and Uncertain Resource Constraints**, April 1988. *Proceedings of the Seventh National Conference on Artificial Intelligence*, pp. 111-116, AAAI '88, Minneapolis, Minnesota in August 1988., April 1988. 6 pages
- KSL 88-37 (Working Paper) Liam Peyton, PhD; **A Transformational Approach to Software Redesign**, May 1988. 23 pages
- KSL 88-39 Anthony Zygmunt; **SOLACE: Systems Optimization Laboratory's Automated Computational Expertise**, May 1988. 55 pages.

- KSL 88-40 Harold P. Lehmann; **Knowledge Acquisition for Probabilistic Expert Systems**. Published in *SCAMC Proceedings*, November 1988. 6 pages.
- KSL 88-41 (Working Paper) Alan C. Noble and Everett C. Rogers; **AIRTRAC Path Association: Development of a Knowledge-Based System for a Multiprocessor**, June 1988. 101 pages
- KSL 88-42 (Working Paper) Alan C. Noble; **ELMA Programmers Guide**, August 1988. 36 pages.
- KSL 88-45 Susan M. Altman; **Representing and Editing Constraints: A Case Study in Financial Resource Management**, June 1988. 42 pages.
- KSL 88-47 R. Martin Chavez and Gregory F. Cooper; **KNET: Integrating Hypertext and Normative Bayesian Modeling**, June 1988. 8 pages
- KSL 88-50 Barbara Hayes-Roth, Micheal Hewett, Richard Washington, Rattikorn Hewett, Adam Seiver; **Distributing Intelligence Within an Individual**, October 1988. To appear in : *Distributed Artificial Intelligence*, Vol 2, L. Gasser and M.N. Huhns, (eds.), Morgan Kaufman, 1988. 23 pages
- KSL 88-52 (Working Paper) John Sullivan; **RL3: An Approach to Incremental Rule Learning**, June 1988. 24 pages
- KSL 88-57 Richard M. Keller; **Learning Approximate Concept Descriptions**, July 1988. 15 pages
- KSL 88-58 David M. Combs, Samson W. Tu, Mark A. Musen, Lawrence M. Fagan; **From Expert Models to Expert Systems: Translation of an Intermediate Knowledge Representation**, August 1988, 13 pages
- KSL 88-59 Mark A. Musen and Johan Van der Lei; **Of Brittleness and Bottlenecks Challenges in the Creation of Pattern-Recognition and Expert-System Models**, August 1988. 18 pages.
- KSL 88-60 Thierry Barsalou, M.D.; **An Object-Based Architecture for Biomedical Expert Database Systems**, August 1988. Published in *SCAMC Proceedings*, November 1988. 8 pages.
- KSL 88-61 Edward H. Shortliffe, M.D., Ph.D.; **Medical Knowledge and Decision Making**, September 1988. 14 pages.
- KSL 88-62 Max Hailperin; **Load Balancing for Massively-Parallel Soft Real-Time Systems**, August 1988. 19 pages

- KSL 88-63 Curtis P. Langlotz and Edward H. Shortliffe; **An Analysis of Categorical and Quantitative Methods for Planning under Uncertainty**, September 1988. Published in *SCAMC Proceedings*, November 1988. 6 pages.
- KSL 88-64 Barbara Hayes-Roth; **Making Intelligent Systems Adaptive**, September 1988, 24 pages. Also appears in K. VanLehn (Ed.) *Architectures for Intelligence*. Lawrence Erlbaum, 1989.
- KSL 88-66 Penny Nii, Nelleke Aiello, James Rice, **Experiments on Cage and Poligon: Measuring the Performance of Parallel Blackboard Systems**, to appear in *Distributed Artificial Intelligence II*. Pitman Publishing Ltd. & Morgan Kaufmann, 1989. February 1989, 69 pages.
- KSL 88-69 James Rice, **The Elint Application on Poligon: The Architecture and Performance of a Concurrent Blackboard System**, December 1988. 11 pages.
- KSL 88-70 Michael a. Shwe, Samson W. Tu, Lawrence M. Fagan; **Validating the Knowledge Base of a Therapy-Planning System**, published in: *Methods of Information in Medicine 28(1): 36-50*, 1989., April 1989. 16 pages.
- KSL 88-71 James Rice; "The Advanced Architectures Project". March 1989. 27 pages.
- KSL 88-73 Mark A. Musen. **Generation of Visual Languages for Development of Knowledge-Based Systems**. Chapter in *Visual Languages*, Volume II (R. R. Korfhage, E. Jungert, and T. Ichikawa, eds.) , New York: Plenum, 1989. 26 pages.
- KSL 88-74 Beverley Kane and Donald W. Rucker; **AI in Medicine**. *AI Expert*, pp. 48-55, November 1988. 9 pages.
- KSL 88-75 Holly B. Jimison; **A Representation for Gaining Insight into Clinical Decision Models**, November 1988. 5 pages.
- KSL 88-76 Leslie Lenert, Lewis Sheiner, Terrence Blaschke; **Improving Drug Dosing in Hospitalized Patients: Automated Modeling of Pharmacokinetics for Individualization of Drug Dosing Regimens** Published in *SCAMC Proceedings*, November 1988. 6 pages
- KSL 88-77 Homer L. Chin; **Case-Based Tutoring from a Medical Knowledge Base**. Published in *SCAMC Proceedings*, November 1988. 8 pages.
- KSL 88-80 Nelleke Aiello; **Cage: The Performance of a Concurrent Blackboard Environment**. December 1988. 11 pages.

- KSL 88-81 Gregory T. Byrd, Nakul P. Saraiya, and Bruce Delagi.
Multicast Communication in Multiprocessor Systems.
Submitted for publication to: *1989 International Conference on Parallel Processing*, January 1989. 19 pages.
- KSL 88-82 Edward H. Shortliffe; **Testing Reality: The Introduction of Decision-Support Technologies for Physicians**, published as an editorial in *Methods of Information In Medicine*, 28: 1-5, 1989. 6 pages.
- KSL 88-83 Edward H. Shortliffe, Lawrence M. Fagan; **"Research Training in Medical Informatics: The Stanford Experience."**
Proceedings of the International Symposium on Medical Informatics and Education, Victoria, B.C., May 1989. March 1989. 8 pages.
- KSL 88-84 I. A. Beinlich, H.J. Suermondt, R.M. Chavez and G.F. Cooper.
The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks.
Submitted to *AI in Medicine*, London 1989.
- KSL 88-85 Nakul P. Saraiya. **A Shared Memory Lisp Package for CARE** January 1989. 7 pages.
- KSL 88-86 Perry L. Miller and Glenn D. Rennels; **Prose Generation from Expert Systems. An Applied Computational Approach**, November 1988. *AI Magazine*, pp. 37-44, Fall 1988. 9 pages.
- KSL 89-01 Eric J. Horvitz, David E. Heckerman and Gregory Cooper;
Reflection and Action Under Scarce Resources: Theoretical Principles and Empirical Study. To appear in: *IJCAI-89*, May 1989. 17 pages.
- KSL 89-02 Rich A. Acuff; **Performance of Two Common Lisp Programs on Several Systems.** January 1989. 30 pages.
- KSL 89-03 Thierry Barsalou, R. Martin Chavez, Gio Wiederhold.
Hypertext Interfaces for Decision-Support Systems: A Case Study. Submitted to: *Medinfo 89*. 6 pages.
- KSL 89-04 Geoffrey Rutledge, George Thomson, Ingo Beinlich, Brad Farr, Michael Kahn, Lewis Sheiner, and Lawrence Fagan.; **VentPlan: An Architecture for Combining Qualitative and Quantitative Computation**, to appear in *IJCAI-89*, January 1989. 9 pages.
- KSL 89-05 Barbara Hayes-Roth, Rich Washington, Rattikorn Hewett, Micheal Hewett, and Adam Seiver; **Intelligent Real-Time Monitoring and Control**, January 1989 15 pages. And in *Proceedings of the International Joint Conference on Artificial Intelligence*, IJCAI-89, Detroit, MI., 1989.

- KSL 89-06 Richard Washington, Barbara Hayes-Roth; **Data Management in Real-Time AI Systems.** March 1989. 11 pages. And in *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-89, Detroit, MI, 1989.*
- KSL 89-07 Mark A. Musen; **Languages for Knowledge Acquisition: Building and Extending Models,** January 1989. *Proceedings of AAAI Spring Symposium on Knowledge System Development Tools and Languages, Stanford, CA March 1989.* 7 pages.
- KSL 89-08 James Rice and Nelleke Aiello; **"See How They Run. The Architecture and Performance of Two Concurrent Blackboard Systems."** To appear in: **Blackboard Architectures and Applications: Current Trends.** March 1989. 22 pages.
- KSL 89-09 Mark A. Musen; **Widening the Knowledge-Acquisition Bottleneck: Automated Tools for Building and Extending Clinical Methods.** *AAMSI Proceedings.* February 1989. 7 pages.
- KSL 89-10 Peter D. Karp, David C. Wilkins; **An Analysis of the Distinction Between Deep and Shallow Expert Systems.** To appear in: *International Journal of Expert Systems, 1989.* February 1989. 36 pages.
- KSL 89-13 Harold P. Lehmann, M.D; **Review of the Uncertainty in AI Workshops.** April 1989. 24 pages.
- KSL 89-14 R. Martin Chavez; **Hypermedia and Randomized Algorithms for Medical Expert Systems.** Submitted to *SCAMC 89.* March 1989. 19 pages.
- KSL 89-15 Gregory T. Byrd and Bruce A. Delagi; **Support for Fine-Grained Message Passing in Shared Memory Multiprocessors.** To appear in: *Proceedings of the 5th Annual Computer Science Symposium, University of South Carolina, April 7-8, 1989.* 21 pages.
- KSL 89-16 Nakul P. Saraiya, Bruce A. Delagi and Sayuri Nishimura, **Design and Performance Evaluation of a Parallel Report Integration System** Submitted for publication. April 1989. 23 pages.
- KSL 89-17 H.J. Suermondt and G.F. Cooper; **Probabilistic Inference in Multiply Connected Belief Networks Using Loop Cutsets.** Submitted to: *International Journal of Approximate Reasoning.* March 1989. 33 pages.
- KSL 89-18 Thierry Barsalou and Gio Wiederhold; **A Cooperative Hypertext Interface to Relational Databases.** Submitted to *SCAMC '89.* March 1989. 22 pages

- KSL 89-19 D.W. Rucker and E.H. Shortliffe; **A Methodology for Implementing Clinical Algorithms Using Expert System and Database Tools.** Submitted to *SCAMC '89*. March 1989.
- KSL 89-20 Brad R. Farr; **Decision-Theoretic Evaluation of Therapy Plans.** Submitted to *SCAMC '89*. March 1989. 10 pages.
- KSL 89-21 John Reed; **Building Decision Models that Modify Decision Systems.** Submitted to *SCAMC '89*. March 1989. 18 pages.
- KSL 89-23 H.L. Suermondt and M.D. Amylon, M.D.; **Probabilistic Prediction of the Outcome of Bone-Marrow Transplantation.** Submitted to *SCAMC '89*. March 1989. 9 pages.
- KSL 89-24 E.J. Horvitz, D.E. Heckerman, K.C. Ng, B.N. Nathwani; **Heuristic Abstraction in the Decision-Theoretic Pathfinder System.** Submitted to *SCAMC '89*. March 1989. 25 pages.
- KSL 89-25 D.E. Heckerman, E.J. Horvitz, B.N. Nathwani; **Toward Effective Normative Decision Systems: Update on the Pathfinder Project.** Submitted to *SCAMC '89*. March 1989. 26 pages.
- KSL 89-26 Mark A. Musen; **Automated Support for Building and Extending Expert Models.** To appear in: a special issue of *Machine Learning*. May 1989. 29 pages.
- KSL 89-28 Mark A. Musen and Johan van der Lei; **Knowledge Engineering for Clinical Consultation Programs: Modeling the Application Area.** Published in *Methods of Information in Medicine*, 28-28-35, 1989. March 1989 9 pages.
- KSL 89-31 R. Martin Chavez and Gregory F. Cooper; **An Empirical Evaluation of a Randomized Algorithm for Probabilistic Inference.** Published in: *Fifth Workshop on Uncertainty in Artificial Intelligence*, April 1989, 13 pages.
- KSL 89-33 Bruce G. Buchanan and Edward H. Shortliffe; **Advances in Expert Systems (A White Paper and Commentary).** Presented to *DARPA, February, 1989*. April 1989. 21 pages.
- KSL 89-34 Michael G. Kahn, Lawrence M. Fagan, and Lewis B. Sheiner; **Model-Based Interpretation of Time-Varying Medical Data.** April 1989. 25 pages
- KSL 89-41 Harold P. Lehmann; **A Decision-Analytic Model for Using Scientific Data,** submitted to *AAAI Workshop in Uncertainty in AI*, May 1989. 19 pages.
- KSL 89-42 E.J. Horvitz, H.J. Suermondt, G.F. Cooper; **Bounded Conditioning: Flexible Inference for Decisions Under Scarce Resources.** May 1989. 21 pages.

Other Publications Not Yet in the KSL Report Series

- 1) Hayes-Roth, B., Hewett, M., Washington, R., Hewett, R., and Seiver, A. **Distributing intelligence within a single individual.** In L. Gasser and M.N. Huhns (Eds.) *Distributed Artificial Intelligence* Volume 2. Morgan Kaufmann, 1989.
- 2) Hewett, R., and Hayes-Roth, R. **Representing and reasoning about physical systems using generic models.** In J. Sowa (Ed.) *Formal Aspects of Semantic Networks*. Morgan Kaufmann, 1989.
- 3) Hayes-Roth, B. **Dynamic control planning in adaptive intelligent systems.** *Proceedings of the DARPA Knowledge-Based Planning Workshop*, 1989.

III.A.2.6. Resource Equipment

The SUMEX-AIM resource is a complex, integrated facility comprised of mainframes, servers, workstations, and networks illustrated in Figures 2 - 8. A key role of the SUMEX-AIM resource is to continue to evaluate workstations in order to keep up with the rapidly changing technology. This evaluation includes new hardware and software, 1) to provide superior development and execution platforms for AI research, 2) to experiment with systems practical for the dissemination of AI systems and their integration with other biomedical systems, and 3) to support the ancillary "office environment" (presently carried out on the SUN-4 and Macintosh's, following the phase-out of the DEC 2060). Thus far no single workstation has materialized that provides all the services we would like to see in support of these missions. This means that for the foreseeable future, we will utilize a multiplicity of machines and software to address the needs of AIM projects.

Systems based on the Motorola 68030 chip (e.g., Apple Macintosh II or NeXT workstations), the Intel 80286 and 80387 chips (e.g., IBM PS/1-4 machines), and other reduced instruction set computer (RISC) chips (e.g., the SUN SPARC or MIPS R-2000 chips), have Lisp benchmark data rivaling the performance of existing, specially microcoded Lisp machines (see Appendix B). It is still early to predict how this "race" will ultimately turn out and software environments play an equally important role with raw hardware speed in the decision. For now, the Lisp software environments on the "stock" machines are not nearly so extensively developed as on Lisp machines and conversely, the routine computing environments of Lisp machines (text processing, mail, spreadsheets, etc.) lag the tools available on stock UNIX machines.

We had been seeking an integration of both the Lisp machine and stock machine worlds. As discussed extensively in the progress section on Core Systems Research, these two capabilities came together as never before with the Macintosh II and microExplorer coprocessor systems.

(1) Purchases This Past Year

The relatively small amount of SUMEX-AIM money for new equipment purchases has been concentrated on upgrades to facilitate the move to the SUN-4 environment, experimental workstations, and server equipment needed for distributed system development. These purchases are paced carefully with the developments of higher performing, more compact, and lower cost systems. The NIH-funded purchases this past year are summarized below. Note that the very large purchase of Mac II workstations, TI microExplorer coprocessors, and the SUN-4 network server was funded almost entirely from *non-NIH* funding sources, even though a significant part of this equipment benefits the AIM community as a whole. Again this year, numerous purchases were made by Stanford research projects to support their work from non-NIH sources.

(#)	<u>Device Purchased</u>	<u>Cost</u>	<u>Comments</u>
(3)	Mac II CPU's, with 24" monitors, 20 MB disks, keyboards, and AppleNet connections.	\$12,400	These machines were to finish outfitting staff desks with Mac workstations.
(2)	Mac II Ethernet boards	\$900	Direct Ethernet connection boards for systems development work.
(1)	ImageWriter LQ printer	\$1,000	Experimental, inexpensive, color impact printer
(3)	LaserWriter II NTX printers	\$11,500	Printer upgrades to replace unreliable IMAGEN 12/300 printers that could not be made to process PostScript.
(36)	2 MB memory expansion kits - Macs	\$21,200	Mac II memory upgrades to allow running MultiFinder and large programs (e.g., HyperCard).
(4)	Trailblazer stand-alone modems	\$4,000	Experimental high-speed modems for remote Ethernet connections.
(4)	Multibus Ethernet Controllers	\$5,200	Upgrades for key Ethernet gateway to prevent lost packets during heavy traffic.
(1)	SUN disk controller	\$3,300	SUN-4 disk controller upgrade to increase performance.
(1)	140 Meg hard drive	\$1,300	Mac II external disk to facilitate systems support and maintenance.
(2)	NeXT machines w/ 330 meg drives	\$18,000	Experimental workstations to evaluate the hardware design and Interface Builder software tools.
(1)	Helical scan back-up subsystem	\$3,400	Experimental high-density tape backup system for large file server support
(5)	Pagers for key systems personnel	\$2,000	For improved communications with staff to handle system emergencies.

(1) Cisco processor- 10 MHz	\$2,000	Upgrade to EtherTIP system so we can run standard commercial TIP software.
(1) Ricoh fax machine	\$400	1/4 share of a jointly used FAX machine to improve communications with AIM community and other contacts
Total Equipment Cost	\$86,600	

(2) Current Subsystem Configurations

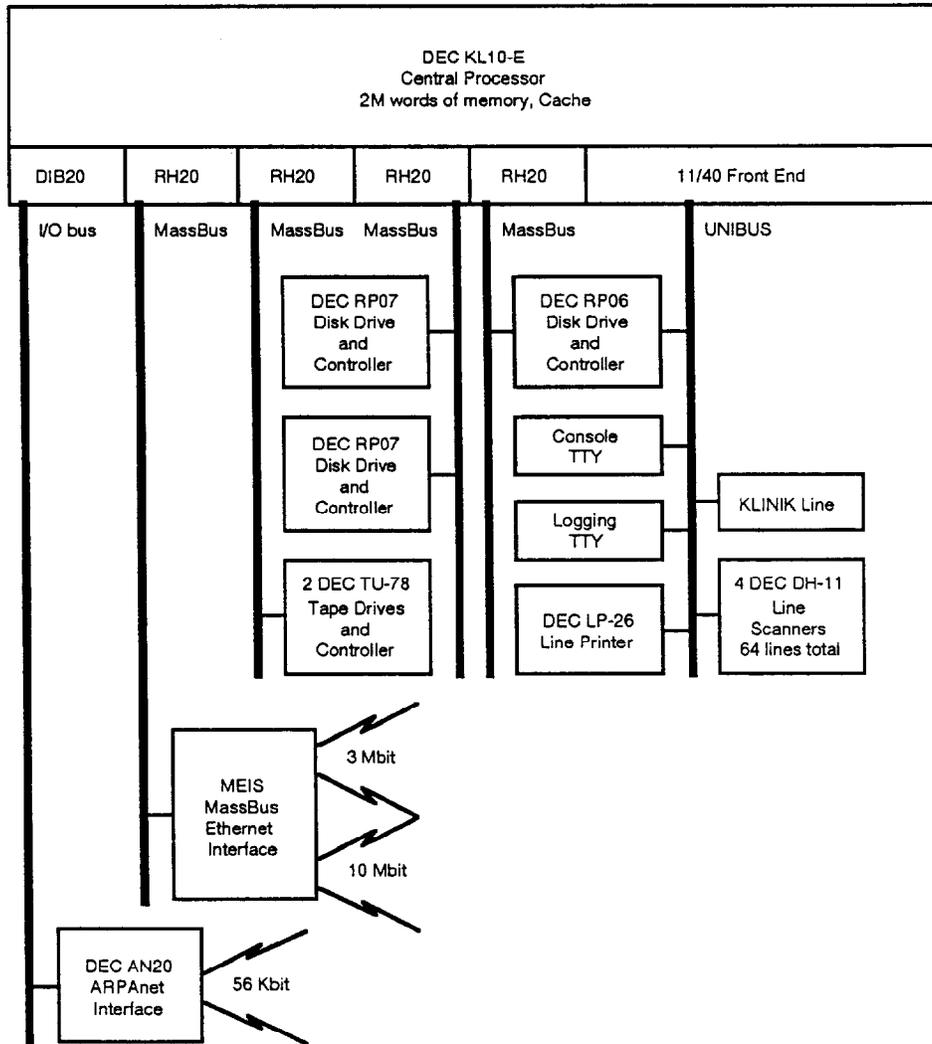


Figure 2. SUMEX-AIM DEC 2060 Configuration

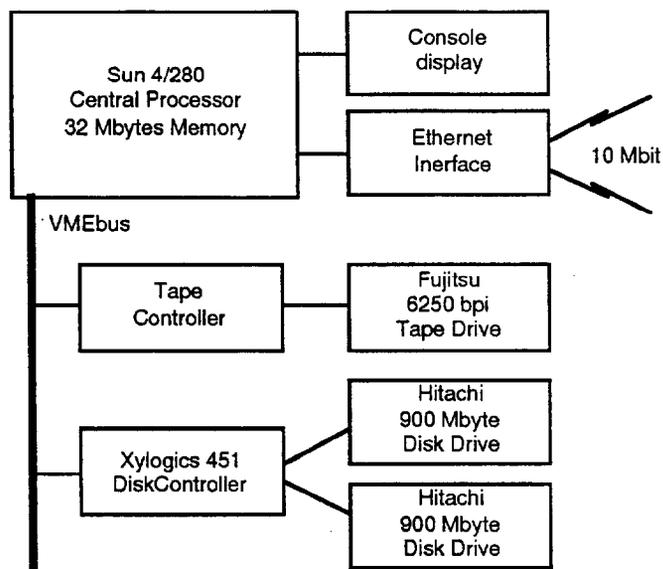


Figure 3. SUMEX-AIM SUN-4 Configuration

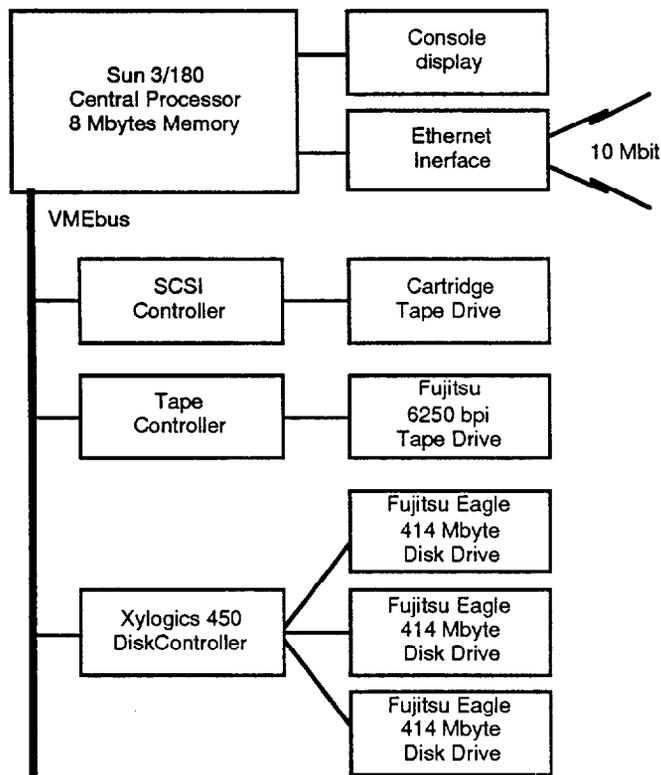


Figure 4. SUMEX-AIM SUN-3 File Server Configuration

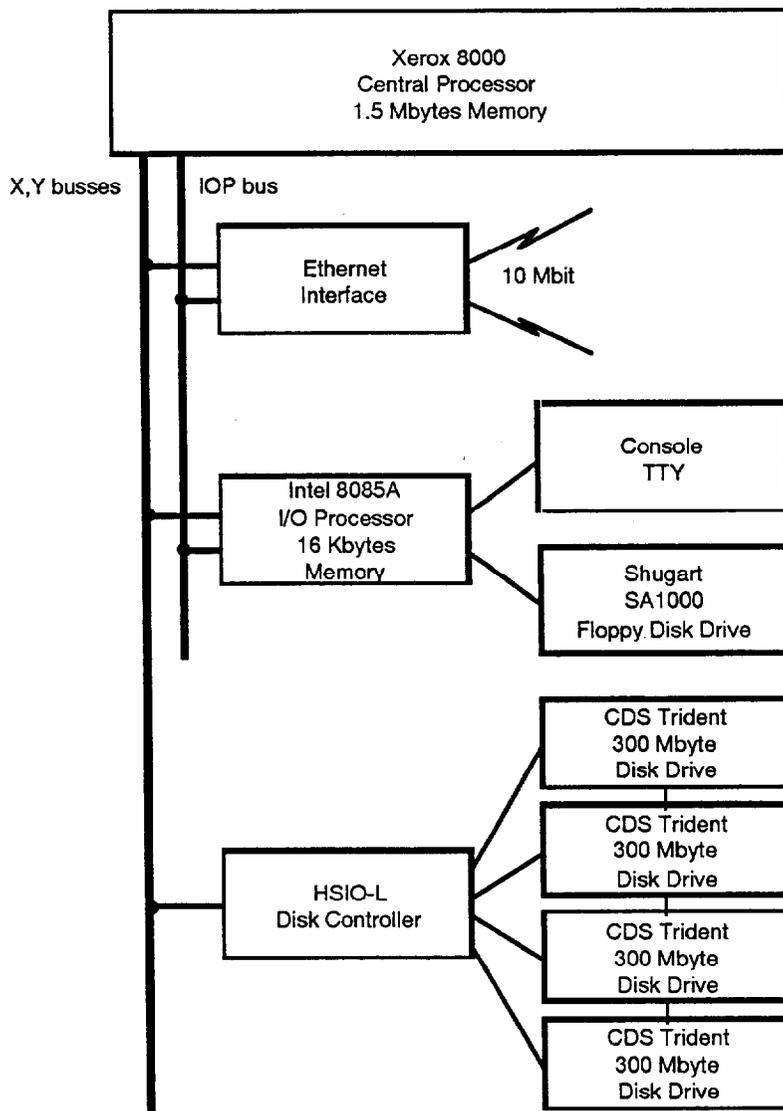


Figure 5. SUMEX-AIM Xerox File Server Configuration

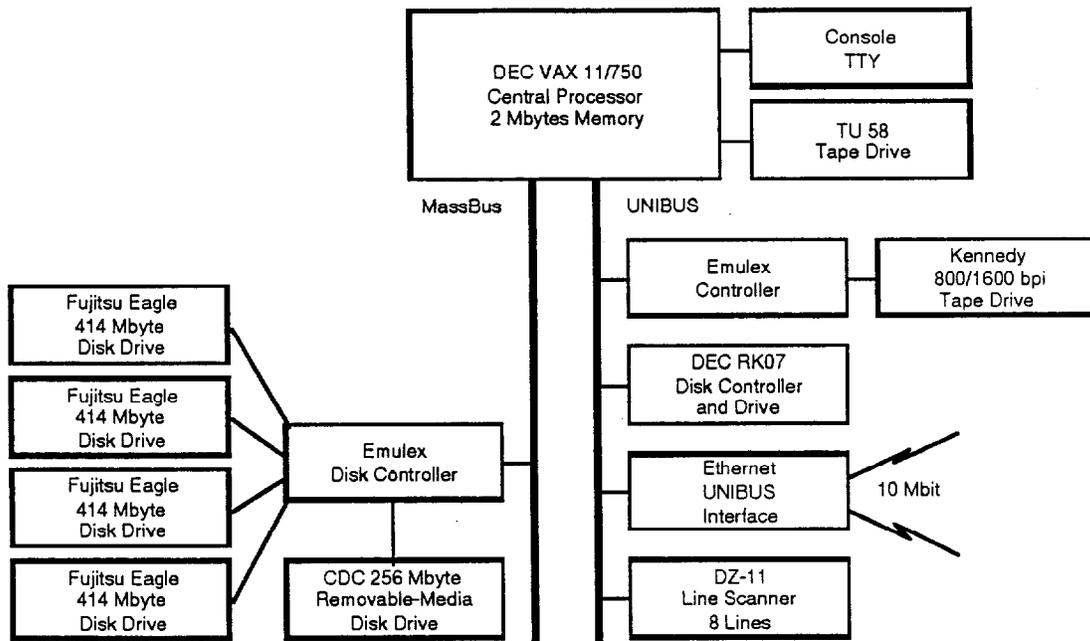


Figure 6. SUMEX-AIM VAX File Server Configuration

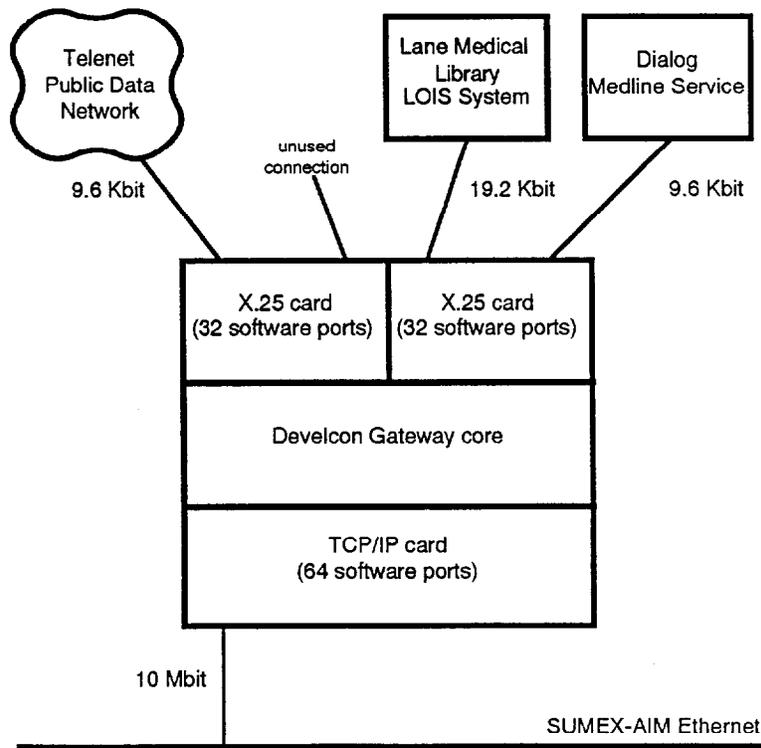


Figure 7. SUMEX-AIM Develcon X.25/TCP-IP Gateway Configuration

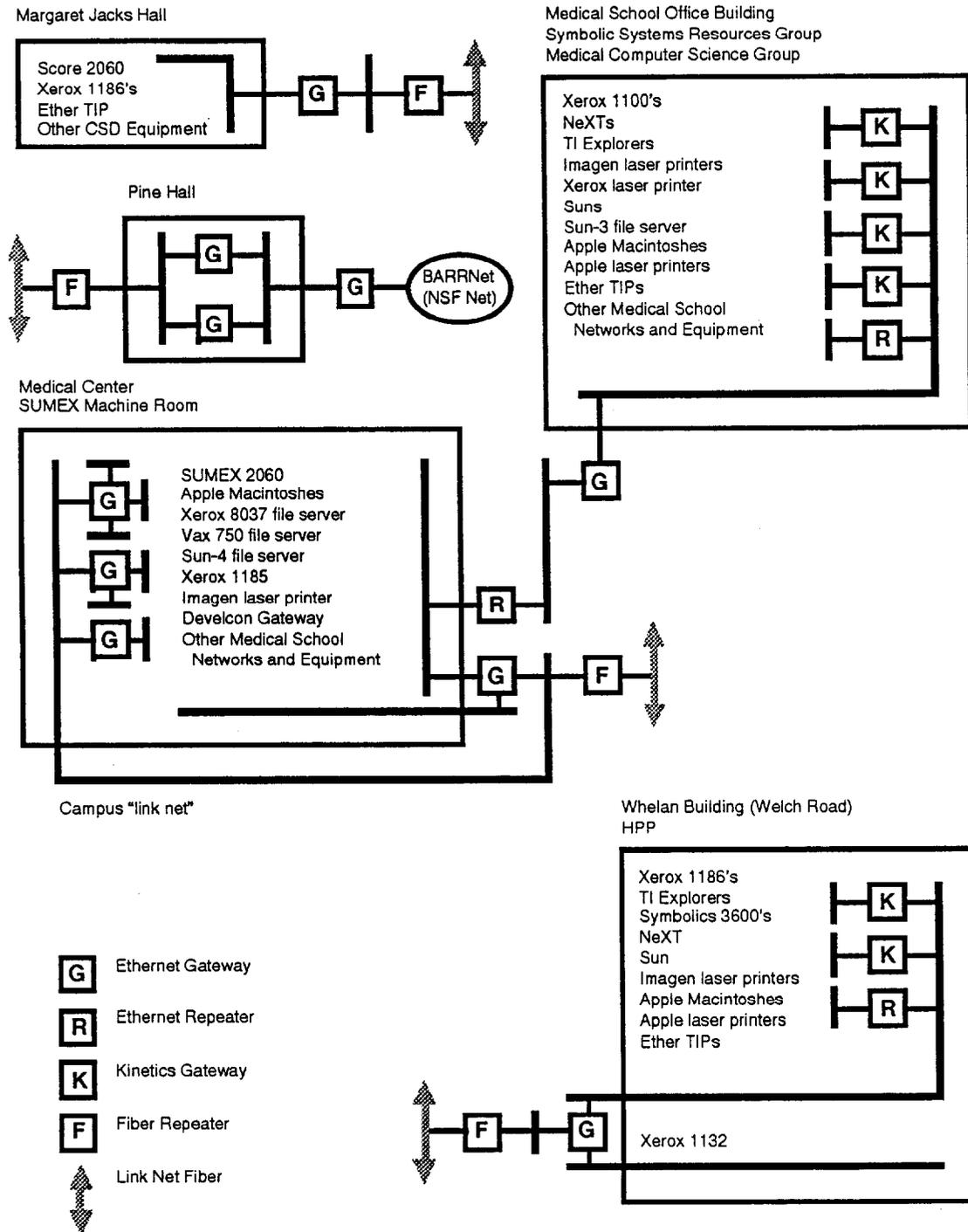


Figure 8. SUMEX-AIM Ethernet Configuration

III.A.2.7. Training Activities

The SUMEX resource exists to facilitate biomedical artificial intelligence applications. This user orientation on the part of the facility and staff has been a unique feature of our resource and is responsible in large part for our success in community building. The resource staff has spent significant effort in assisting users to gain access to the SUMEX-AIM resources at Stanford and use it effectively as well as in assisting AIM projects in designing their own local computing resources based on SUMEX experience. We have also spent substantial effort to develop, maintain, and facilitate access to documentation and interactive help facilities. The HELP and Bulletin Board subsystems have been important in this effort to help users get familiar with the computing environment.

We have regularly accepted a number of scientific visitors for periods of several months to a year, to work with us to learn the techniques of expert system definition and building and to collaborate with us on specific projects. Our ability to accommodate such visitors is severely limited by space, computing, and manpower resources to support them within the demands of our on-going research.

Finally, the training of graduate students is an essential part of the research and educational activities of the KSL. Based largely on the SUMEX-AIM community environment, we have had two unique, special academic degree programs at Stanford, the Medical Information Science program and the Masters of Science in AI, to increase the number of students we produce for teaching, research, and industry. A number of students are pursuing interdisciplinary programs and come from the Departments of Engineering, Mathematics, Education, and Medicine.

The *Medical Information Sciences* (MIS) program continues to be one of the most obvious signs of the local academic impact of the SUMEX-AIM resource¹. The MIS program received University approval (in October 1982) as an innovative training program that offers MS and PhD degrees to individuals with a career commitment to applying computers and decision sciences in the field of medicine. In Spring 1987, a University-appointed review group unanimously recommended that the degree program be continued for another five years. The MIS training program is based in the School of Medicine, directed by Dr. Shortliffe, co-directed by Dr. Fagan, and overseen by a group of six University faculty that includes two faculty from the Knowledge Systems Laboratory. The specialized curriculum offered by the new program is intended to overcome the limitations of previous training options. It focuses on the development of a new generation of researchers with a commitment to developing new knowledge about optimal methods for developing practical computer-based solutions to biomedical needs.

¹ Shortliffe, E. H., and Fagan, L. M. Research Training in Medical Informatics: The Stanford Experience. In Proceedings of *The International Symposium on Medical Informatics and Education*, Victoria, B.C., 1989.

The program accepted its first class of four trainees in the summer of 1983 and has now reached its steady-state size of approximately twenty-four graduate students. The program encourages applications from any of the following:

- medical students who wish to combine MD training with formal degree work and research experience in MIS;
- physicians who wish to obtain formal MIS training after their MD or their residency, perhaps in conjunction with a clinical fellowship at Stanford Medical Center;
- recent BA or BS graduates who have decided on a career applying computer science in the medical world;
- current Stanford undergraduates who wish to extend their Stanford training an extra year in order to obtain a "co-terminus" MS in the MIS program;
- recent PhD graduates who wish post-doctoral training, perhaps with the formal MS credential, to complement their primary field of training.

In addition, a special one-year MS program is available for established academic medical researchers who may wish to augment their computing and statistical skills during a sabbatical break. As of spring of 1989, 46% of our enrolled trainees have previously received MD degrees and another 25% are medical students enrolled in joint degree programs. About 29% are candidates for the MS degree, while the remaining 71% are doctoral students. The program has 12 graduates to date, and another 4 students are expecting to complete degrees in June of 1989.

Except for the special one-year MS mentioned above, all students spend a minimum of two years at Stanford (four years for PhD students) and are expected to undertake significant research projects for either degree. Research opportunities abound, however, and they of course include the several Stanford AIM projects as well as research in psychological and formal statistical approaches to medical decision making, applied instrumentation, large medical databases, molecular biology, and a variety of other applications projects at the medical center and on the main campus. Several students are already contributing in major ways to the AIM projects and core research described elsewhere in this annual report.

We are pleased that the program already has an excellent reputation and is attracting superb candidates for training positions. The program's visibility and reputation is due to a number of factors:

- high quality students, many of whom publish their work in conference proceedings and refereed journals even before receiving their degrees; Stanford MIS students have won first prize in the student paper competition at the Symposium on Computer Applications in Medical Care (SCAMC) in 1985 and 1986, and have also received awards for their work at annual meetings of organizations such as the Society for Medical

Decision Making, the American Association for Medical Systems and Informatics (AAMSI), and the American Association for Artificial Intelligence (AAAI);

- a rigorous curriculum that includes newly-developed course offerings that are available to the University's medical students, undergraduates, and computer science students as well as to the program's trainees;
- excellent computing facilities combined with ample and diverse opportunities for medical computer science and medical decision science research;
- the program's great potential for a beneficial impact upon health care delivery in the highly technologic but cost-sensitive era that lies ahead.

The program has been successful in raising financial and equipment support from industry and foundations. It is also recipient of a training grant from the National Library of Medicine. The latter grant was recently renewed for another five years with a study section review that praised both the training and the positive contribution of the SUMEX-AIM environment.

III.A.2.8. Resource Operations and Usage

(1) Operations and Support

The diverse computing environment that SUMEX-AIM provides requires a significant effort at operations and support to keep the resource responsive to community project needs. This includes the planning and management of physical facilities such as machine rooms and communications, system operations routine to backup and retrieve user files in a timely manner, and user support for communications, systems, and software advice. Maintaining the quality of these services has become increasingly difficult in the face of recent budget cuts and attendant staff cuts.

We spend significant time on new product review and evaluation such as Lisp workstations, terminals, communications equipment, network equipment, microprocessor systems, mainframe developments, and peripheral equipment. We also pay close attention to available video production and projection equipment, which has proved so useful in our dissemination efforts involving video tapes of our work.

We continue to operate the primary elements of our server equipment in a generally unattended manner. Operations costs are kept to a minimum by utilizing a student staff for routine tasks. Senior members of this staff provide improvements to the operations procedures in addition to training and supervising new students. This has provided SUMEX with a cost effective operations scheme, contributed to the education of the students, and assisted students in meeting their obligations in undergraduate financial aid programs.

While most of our equipment is concentrated in three computer equipment rooms, our move towards distributed computing has resulted in a substantial amount of equipment being installed in offices and student carrels. This physical distribution of the environment means that maintenance tasks (hardware and software) are more time-consuming because of the need to attend to the systems in remote locations.

(2) Resource Usage Details

The following data give the most cursory overview of various aspects of the SUMEX-AIM resource usage, based on the DEC 2060 accounting system in operation for part of the year — from May 1988 until the transition to the SUN-4 system in October and November of 1988. Measuring subsequent usage in a distributed environment is a much more difficult and ambiguous undertaking. It does not make sense, for example, to try to tally CPU or disk usage on individual workstations, anymore than we would try to measure a researcher's use of his pocket calculator. Similarly, much of the usage of the central servers is of a transient nature, copying a file here or there, archiving or retrieving a file, printing a document through the system spooler, reading mail and transmitting replies, etc. In the client/server model of our work environment, most often a user is not even logged in to a server in order to

perform his tasks or if he is, most of the work is going on on his workstation with the central server providing background supporting services. Simply stating the server usage statistics (e.g., in CPU time consumed, connect time, or disk space used) does not begin to approach the measurement of effective computing resource usage in the pursuit of our research goals. To try to do so meaningfully would require a very large "instrumentation" effort that is not even contemplated in distributed workstation systems as they are distributed by vendors or third party system developers, that is not within the Council-approved research goals of the resource, and that would require an additional effort well beyond the resources that are available, with dubious return for the investment.

For example, since we moved our TELENET service to operate through the Develcon X.25/Ethernet gateway, connections from other parts of the country look *exactly* like connections from local workstations since they come into our servers over common-service network connection ports and are assigned randomly-selected TCP sockets for subsequent services. Since the Develcon gateway developer does not provide any accounting tools and the gateway itself does not know *who* the person is who is trying to establish a connection, we have no way to break down network usage between local and remote users, much less identify who is using how much of the various core services.

Thus, one of the consequences of our move to a distributed computing environment is that we will not be able to provide future meaningful data on resource usage as is possible with other central resources. For this year's report, we close out the record for the DEC 2060 usage with the following data:

- Overall resource loading data.
- Individual project and community usage.

(2.1) Overall Resource Loading Data

The following plot (see Figure 9) shows resource CPU usage over the entire history of the SUMEX-AIM project. This includes data from both the KI-TENEX system and the current DECsystem 2060. At the point where the SUMEX-AIM community switched over to the 2060 (February, 1983), you will notice a sharp change in the graphs. This is due to differences in scheduling, accounting, and processor speed calculations between the systems which we have not attempted to normalize.

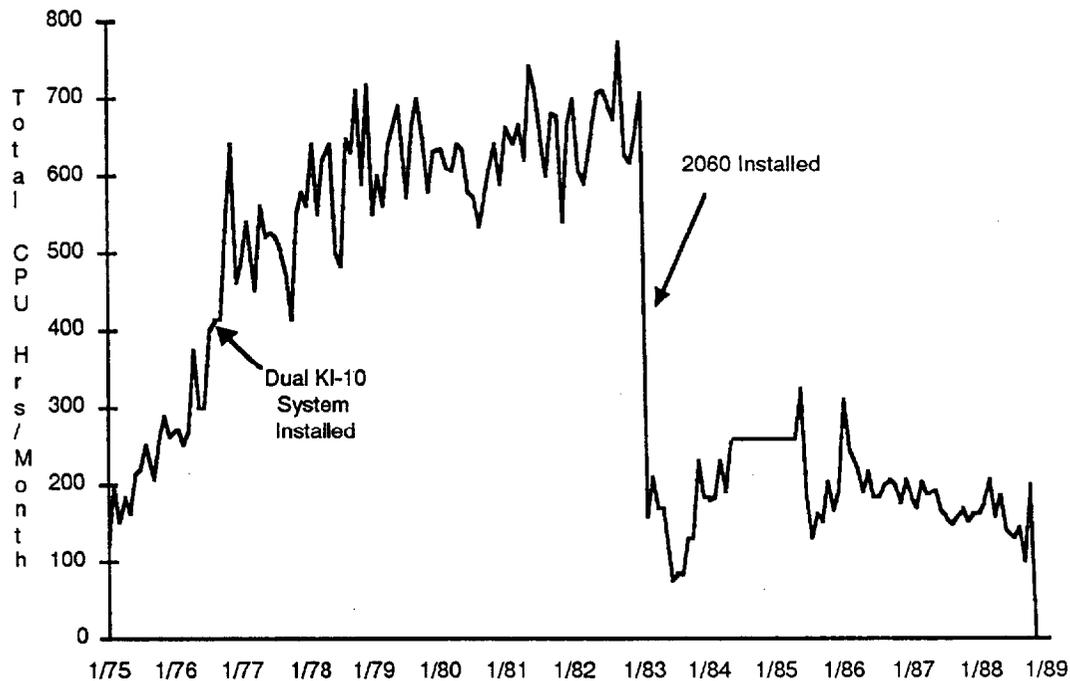


Figure 9. Total CPU Hours Consumed by Month

(2.2) Individual Project and Community Usage

The following histogram (Figure 10) shows the relative central resource usage during the past grant year by national AIM and Stanford collaborative projects and the core research groups before the DEC 2060 phase-out. The bars represent the fraction of total CPU time consumed by each project between May 1, 1988 and October 31, 1988, on the SUMEX-AIM DECsystem 2060 system.

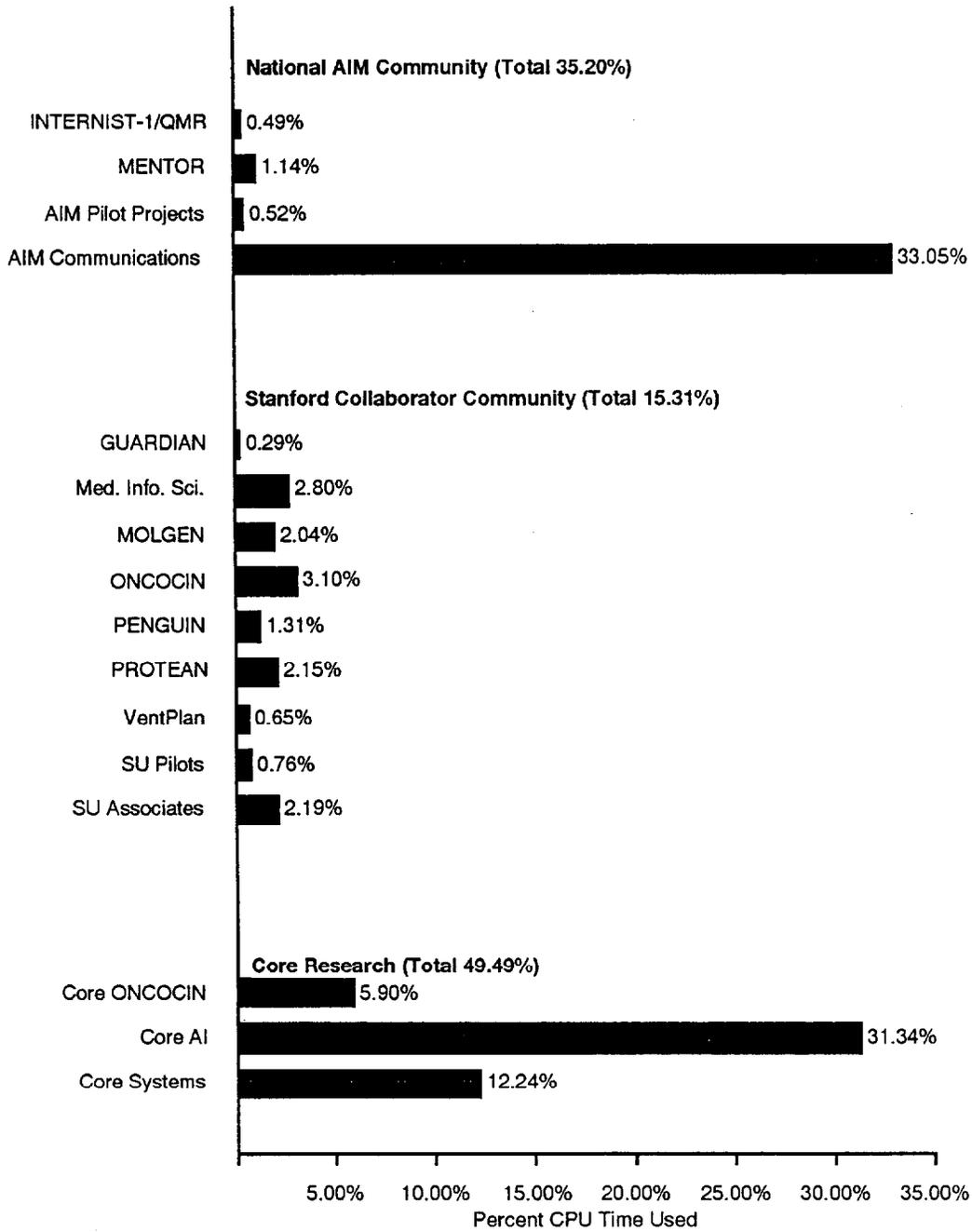


Figure 10. CPU Usage Histogram by Project and Community

Annualized Resource Use by Individual Project - 5/88 through 4/89

The following tabulated form of the data (Figure 12) shows total CPU consumption by project (Hours), total terminal connect time by project (Hours), and average file space in use by each project (Pages, 1 page = 512 computer words). These data were accumulated for each project for the months between May 1988 and October 1988 (when the users were transferred to the SUN-4 server), and were then annualized to the estimated project usage for an entire year.

National AIM Collaborator Community	CPU (Hours)	Connect (Hours)	File Space (Pages)
1) INTERNIST-I/QMR Project Jack D. Myers, M.D. Randolph A. Miller, M.D. University of Pittsburgh	4.20	158	743
2) MENTOR Project <i>Medical Evaluation of Therapeutic Orders</i> Stuart M. Speedie, Ph.D. University of Maryland Terrence F. Blaschke, M.D. Stanford University	9.80	6135	2057
3) AIM Pilot Projects PathFinder (Nathwani and Fagan)	4.44	341	1675
4) AIM Communications			
AIM Mail-Only Users	14.87	4837	8993
AAAI Management	6.29	3099	602
MCS Collaborators	6.76	795	984
MOLGEN Collaborators	1.08	82	1543
File/Information Access	253.22	14312	2
Guest and Other	0.86	45	3188
AIM Community Totals	301.52	29804	19787

Figure 11. Table of Resource Use by Project

Stanford Collaborator Community	CPU (Hours)	Connect (Hours)	File Space (Pages)
1) GUARDIAN Project Barbara Hayes-Roth, Ph.D. Department of Computer Science Adam Seiver, M.D. Department of Surgery Palo Alto VA Hospital	2.52	2141	298
2) Medical Information Sciences Edward H. Shortliffe, M.D., Ph.D. Lawrence M. Fagan, M.D., Ph.D. Department of Medicine	23.99	9436	3595
3) MOLGEN Project <i>Applications of Artificial Intelligence to Molecular Biology: Research in Theory Formation, Testing and Modification</i> Edward A. Feigenbaum, Ph.D. Peter Friedland, Ph.D. Charles Yanofsky, Ph.D. Depts. Computer Science/Biology	17.50	4129	4623
4) ONCOCIN Project <i>Knowledge Engineering for Medical Consultation</i> Edward H. Shortliffe, M.D., Ph.D. Lawrence M. Fagan, M.D., Ph.D. Department of Medicine	26.58	9267	7471
5) PENGUIN Project Gio C.M. Wiederhold, Ph.D. Depts. Computer Science and Medicine	11.24	2956	6315
6) PROTEAN Project Oleg Jardetzky School of Medicine	18.44	4009	2975

Figure 11. Table of Resource Use by Project (Continued)

	CPU (Hours)	Connect (Hours)	File Space (Pages)
Core AI Research			
1) Advanced Architectures Edward A. Feigenbaum, Ph.D. Computer Science Department	85.50	25844	7893
2) Blackboard Architectures Barbara Hayes-Roth, Ph.D. Computer Science Department	46.78	10427	5405
3) Large Multi-Use Knowledge Bases Edward A. Feigenbaum, Ph.D. Richard Keller, Ph.D. Yumi Iwasaki, Ph.D. Computer Science Department	26.62	12428	2019
4) Software Design Project H. Penny Nii Computer Science Department	8.28	1426	1467
5) MS:AI Student Projects Edward A. Feigenbaum, Ph.D. Computer Science Department	5.18	1975	468
6) Machine Learning Studies Bruce G. Buchanan, Ph.D. University of Pittsburgh	20.43	4230	3922
7) SOAR Project Paul R. Rosenbloom, Ph.D. Information Sciences Institute University of Southern California	8.46	6668	563
8) HPP Administration	49.18	15416	9839
9) HPP Associates	18.04	3119	3825
Core AI Research Totals	268.47	81534	35401

Figure 11. Table of Resource Use by Project (Continued)

	CPU (Hours)	Connect (Hours)	File Space (Pages)
Core Systems Research			
1) SUMEX Staff R & D Thomas C. Rindfleisch Departments of Medicine and Computer Science	102.94	29234	17118
2) Systems Associates	1.92	337	1188
Core Systems Research Totals	104.86	29572	18306
	CPU (Hours)	Connect (Hours)	File Space (Pages)
System Operations			
1) System Operations	724.76	85231	2709
2) Other Opns & Mgmt Services	100.22	247	1779
System Operations Totals	824.98	85478	4488
Resource Grand Totals	1681.49	292385	119224

Figure 11. Table of Resource Use by Project (Concluded)

III.B. Research Highlights

In this section we describe several research highlights from the past year's activities. These include notes on existing projects that have passed important milestones, new pilot projects that have shown progress in their initial stages, and other core research and special activities that reflect the progress, impact, and influence the SUMEX-AIM resource has had in the scientific and educational communities.

III.B.1. INTERNIST-I/QMR

The INTERNIST-1/QMR project, under Drs. Jack Myers and Randy Miller at the University of Pittsburgh, has been developing a high-level computer diagnostic program in the broad field of internal medicine as an aid in the solution of complex medical problems. This system has won recognition both as a project in artificial intelligence, encoding one of the largest knowledge bases ever attempted, covering almost 600 different diseases and 4500 possible patient findings and as an able problem-solver in internal medicine.

Over the past decade, this program has been able to analyze many hundreds of difficult diagnostic problems, including cases published in medical journals (particularly Case Records of the Massachusetts General Hospital, in the *New England Journal of Medicine*), CPCs, and unusual problems of patients in the Pittsburgh Medical Center. In most instances, although not all, INTERNIST-I has performed at the level of the skilled internist. The INTERNIST-I program has also been used in recent years to develop patient management problems for the American College of Physician's Medical Knowledge Self-assessment Program.

The QUICK MEDICAL REFERENCE (QMR) program, developed under the leadership of Dr. Miller, incorporates most of the INTERNIST-I knowledge base and diagnostic consultative program, together with much more powerful user interface and query tools than INTERNIST-I ever had. And QMR runs on an IBM PC-AT workstation. QMR has served as a means to distribute the knowledge base to over twenty other academic medical institutions where it can be used as an "electronic textbook" in medical education at all levels — by medical students, residents and fellows, and faculty and staff physicians. This distribution is continuing to expand.

The Pittsburgh group, including researchers, residents in internal medicine, and fellows in medical informatics, is continuing to expand the knowledge base and to polish the QMR diagnostic consultant program. The medical knowledge base continues to grow both in the incorporation of new diseases and in the modification of diseases already profiled so as to include recent advances in medical knowledge.

Limited field trials of QMR were begun in 1987 on the clinical services in internal medicine at the Health Center of the University of Pittsburgh. A "computer-based diagnostic consultation service" was made available to attending physicians and house staff of our two main teaching hospitals. Institutional Review Board (IRB) approval was granted to the service before it was initiated. In the near term, these clinical field trials will be extended to other university health centers which have expressed interest in working with the system.

III.B.2. PathFinder

One of the most difficult areas in surgical pathology is the microscopic interpretation of lymph node biopsies. Most pathologists have difficulty in accurately classifying lymphomas. Several cooperative oncology group studies have documented that while experts show agreement with one another, the diagnosis rendered by a "local" pathologist may have to be changed by expert lymph node pathologists (expert hematopathologists) in as many as 50% of the cases.

The National Cancer Institute recognized this problem in 1968 and created the Lymphoma Task Force which is now identified as the Repository Center and the Pathology Panel for Lymphoma Clinical Studies. The main function of this expert panel of pathologists is to confirm the diagnosis of the "local" pathologists and to ensure that the pathologic diagnosis is made uniform from one center to another. But this approach is only a partial answer to the problem as the panel annually reviews only 1,000 cases whereas more than 30,000 new cases of lymphomas are reported each year.

The PathFinder project, under Dr. Bharat Nathwani of the University of Southern California and M.D./Ph.D. students David Heckerman and Eric Horvitz of the Stanford University Medical Computer Science Group, has been exploring the use of a computer-based diagnostic program that provides advice on over 70 common benign and malignant diseases of the lymph node based on over 100 histologic features. The design of the program was influenced by the architecture of the INTERNIST-1 program, also developed on the SUMEX resource.

Project computer science research is studying formal techniques for decision making under uncertainty, including the assessment and representation of important dependencies among morphologic features and diseases, reasoning about the costs and benefits of alternative information acquisition strategies, the acquisition and use of expert knowledge bases from multiple experts, the customization of the system's reasoning and explanation behaviors to reflect the expertise of the user, and controlling the naturalness of complex formal reasoning techniques. A group of expert pathologists from several centers in the U.S. have showed interest in the program and helped to provide the structure of the knowledge base for the Pathfinder system.

Originally written in Lisp on the SUMEX 2060, PathFinder was converted two years ago to MPW Object Pascal on the Macintosh II. Much of the recent testing and refinement of the knowledge base has been carried out within the Macintosh II environment. The group conducted a study to compare the performance of the system with that of the domain expert. In the evaluation, a community pathologist used the Pathfinder system to analyze a set of difficult cases. Fifty-three cases were selected in sequence from a large library of referrals. The work showed a close correspondence between the behavior of the system and expert decision making.

III.B.3. The Distributed SUMEX-AIM Community

SUMEX-AIM has undergone a major transition this past year, with the movement of users from the tried and true DEC 2060 resource to a new UNIX-based SUN-4. During the inexorable development and maturation of inexpensive and powerful workstations that are now on most researcher's desks, the 2060 has provided us with a link to the past and a stable continuation of essential research, communications, and other network services for the AIM community. But, under recent budget pressures, we were unable to continue a reliable, well-maintained 2060 service and so, in October 1988, we brought a SUN-4 UNIX computer on-line to replace the old machine for most SUMEX-AIM community functions. We have continued to operate the 2060 in background mode under a much lower cost and less responsive maintenance arrangement but even this level of access will be ended and the machine will be shut down completely by the end of July. We are making every effort to ensure continued access, through the SUN-4, to all the files that were archived under TENEX (1975 - 1983) and TOPS-20 (1983 - present). We will also provide continued access to the annual full file system dumps we have kept for each of the past 14 years.

Of course, after we shut the 2060 down, none of the early seminal programs developed by the SUMEX-AIM community will be accessible — including DENDRAL, MYCIN/EMYCIN, INTERNIST-1, SECS, MOLGEN/UNITS, AGE, PARRY, GUIDON, RADIX, CRYSLIS, PUFF-VM, and on and on... As we turn this next page in SUMEX-AIM history, we realize that most other 2060's that could run these programs are rapidly disappearing as well. so an era is indeed passing.

But even as we watch these changes nostalgically, the future is bright. The Apple Macintosh II workstations we chose are performing extremely well as a general computing environment for researchers and staff, the TI Explorer Lisp machines (including the microExplorer Macintosh coprocessor) are a sound base as the near-term high-performance Lisp research environment, and, after considerable effort, the SUN-4 has taken over most of the 2060 functions as the central system network server (network services, file services, printing services, etc.). Initial user response to the introduction of these systems has been overwhelmingly enthusiastic, even though there have been many "rough edges" to be smoothed out along the way toward full systems integration. Our core development work is coming along well for providing remote access between workstations and servers, integrating a solid support of the TCP-IP network protocols, and building a powerful distributed electronic mail system. The new Mac II mail system will be introduced this summer and we believe that it will be a significant quantum improvement over the serial TTY-based systems of the past.

III.B.4. ONCOCIN

ONCOCIN, developed under Drs. Ted Shortliffe and Larry Fagan at Stanford University, is an expert system for clinical oncology, designed for use in managing chemotherapy after a diagnosis has been reached. Because anticancer agents tend to be highly toxic, and because their tumor-killing effects are routinely accompanied by damage to normal cells, the rules for monitoring and adjusting treatment in response to a given patient's course over time are complex and difficult to memorize. ONCOCIN integrates a temporal record of a patient's treatment with an underlying knowledge base of treatment protocols and rules for adjusting dosage, delaying treatment, aborting cycles, ordering special tests, and similar management details. The program uses this knowledge to help physicians with decisions regarding the management of specific patients.

Oncologists use ONCOCIN routinely for recording and reviewing patient data, replacing the conventional recording of data on a paper flowsheet. With its knowledge of the patient's chemotherapy protocol, ONCOCIN then provides assistance by suggesting appropriate therapy at the time that the day's treatment is to be recorded on the flowsheet. Physicians maintain control of the decision and can override the computer's recommendation if they wish. ONCOCIN also indicates the appropriate interval until the patient's next treatment and reminds the physician of radiologic and laboratory studies required by the treatment protocol.

The ONCOCIN Project started in July 1979 and the first version ran on DECSys-10/20 mainframes using character-oriented terminals. Limitations in terminal capabilities to adapt to user interface needs and the high cost of central machines necessitated a reimplementaion of the system on workstations in 1984. In 1986, we placed the Xerox Lisp machine version of ONCOCIN in the Stanford Oncology Day Care clinic. This version is a completely different program from the mainframe in that it uses graphical user interfaces extensively, new protocols are entered through a powerful graphical data entry interface (OPAL), and it has a revised knowledge representation and reasoning component. In 1987, we began to explore the use of continuous speech recognition as an alternate entry method for communicating with ONCOCIN.

Although we have successfully moved ONCOCIN into a stable and useful system on the Xerox Lisp workstations, it is now clear that this environment will not provide the means for dissemination we need for economical and technical reasons. Based on the success of this earlier work, strong interest has developed, from such diverse quarters as the National Cancer Institute and the Stanford Hospital, for developing a fully operational version of ONCOCIN that can be broadly used in oncology clinics outside our research laboratory. This past year, the Stanford Hospital started a program to assist in the transfer of innovative medical technology out of the laboratory to patient care and ONCOCIN was selected as one of 10 projects to be funded from a large group of competing proposals. So it is once again time to

consider a redesign of the system. The dilemma for the project that is still unresolved is how to maintain a cohesiveness between ongoing research work to extend ONCOCIN and generalize it for applications to other domains and the operational needs of a widely disseminated practical system. Much thought has gone into this problem this past year, including issues such as which of the modern workstation alternatives to select (Lisp machine, IBM PC, Apple Macintosh, SUN or NeXT UNIX workstation, ...), what language to pick (C, Lisp, ...), and whether the research and operational systems can really be consistent versions of a single system?

In order to understand the scope and practical issues involved, we have begun an experiment to port ONCOCIN to a TI microExplorer running inside of a Mac II during the last six months. We have completed the translation of the Ozone object-oriented system, the temporal network and most of the reasoner. We will next approach the design of the user interface, which must be rewritten anew, since the current interface depends heavily on the graphical capabilities of the Xerox workstations. We are also starting a study of the overall design and specification of an "integrated" oncologist's workstation, under NCI sponsorship, that will lead to an attempt to coordinate federal, academic, and industrial efforts to implement such a system.

III.C. Administrative Changes

There have been few administrative changes within the project this past reporting year. As we reported last year, Professor Buchanan, who had been one of the leaders of the core AI research effort, left in July 1988 to take a faculty post at the University of Pittsburgh. At about the same time, two Research Associates joined the core AI research effort, Dr. Yumi Iwasaki, who finished her PhD at Carnegie Mellon University, and Dr. Thomas Gruber, who did his PhD at the University of Massachusetts. Overall, these changes have caused a shift in emphasis of the core AI research work away from the machine learning work that Prof. Buchanan headed and toward the large, multiuse knowledge base work that has been getting underway.

Effective August 1, 1988, we discontinued the fee-for-service cost recovery system we have been using during the last two years to collect from Stanford users the resource operating costs not covered by NIH support. The operating cost shortfall is now being paid entirely by KSL core research projects (i.e., none of the Stanford or national collaborator projects pays anything for their increasingly communication-oriented use of SUMEX-AIM) — under the previous cost center model, *all* Stanford users paid fees. These direct payments avoid the large accounting overhead of a cost center to collect relatively small bills each month and are now made on the basis of a "bulk purchase" agreement, reached between resource management and the KSL Principal Investigators. Our reasons for this move are described in more detail in section III.D.2 (Cost Center Management).

III.D. Resource Management and Allocation

III.D.1. Overall Management Plan

Early in the design of the SUMEX-AIM resource, an effective management plan was worked out with the Biotechnology Resources Program (now Biomedical Research Technology Program) at NIH to assure fair administration of the resource for both Stanford and national users and to provide a framework for recruitment and development of a scientifically meritorious community of application projects. This structure has been described in some detail in earlier reports and is documented in our recent renewal application. It has continued to function effectively as summarized below.

The AIM Executive Committee meets periodically by teleconference to advise on new user applications, discuss resource management policies, plan workshop activities, and conduct other community business. The Advisory Group meets as needed to review project applications. (See Appendix C for a current listing of AIM committee membership).

We actively recruit new application projects and disseminate information about AI in biomedicine. With the development of more decentralized computing resources within the AIM community outside of Stanford, the use of SUMEX resources by AIM members has shifted more and more toward communication with colleagues and access to information.

With the advice of the Executive Committee, we have opened SUMEX-AIM resources widely to biomedical users desiring electronic communications facilities.

We have carefully reviewed on-going projects with our management committees to maintain a high scientific quality and relevance to our biomedical AI goals.

We continue to provide active support for the AIM workshops. The most recent one was held in the spring of 1988 at Stanford University, under the auspices of the American Association for Artificial Intelligence (AAAI). Planning is underway for an AIM workshop in the spring of 1990.

We have continued to provide systems advice to users attempting to set up computing resources at their own sites, based on the expertise developed in the SUMEX resource environment.

We have tailored resource policies to aid users whenever possible within our research mandate and available facilities.

III.D.2. Cost Center

Our plan for the term of the current grant had been a resolute but responsible transition of the SUMEX-AIM resource to a distributed community model of operation. While there has continued to be a group of

national and local users — particularly young projects needing seed support prior to obtaining major funding — that depend on a central shared resource like the SUMEX mainframe, powerful and widely available workstation equipment has rapidly become accessible at a cost that most projects can afford, even young ones. Thus, the period of critical dependence on the DEC 2060 for raw computing cycles is largely past and its role in supporting routine computing and communication services was soon to be replaced by other more cost effective equipment. We were in the process of implementing the phase-out of the SUMEX 2060 machine over this year when the 11% budget cut forced us to shut down the machine to users in a much more precipitous manner (see the report of Core Systems Development in Section III for more details). In the course of this much more hasty change-over from the DEC 2060-based to a SUN-4-based resource, it became clear that the cost center fee-for-service approach to recovering unsubsidized operating costs was counterproductive to our efforts. There were three main reasons for this assessment:

- In the face of such a large budget cut, the administrative overhead costs associated with running the cost center became prohibitive. Since they would have had to be recovered through the rates charged, this would have raised rates relatively much faster than previously planned, and our user projects were unprepared for this magnitude of increase.
- The rapidly rising costs of fee-for-service computing were having a strongly negative effect on the research we were striving to support — particularly student computing. Students were receiving increasing pressure from their mentors and sponsors to contain costs and so were confronted with unworkable decisions about how to avoid extra computer runs to polish papers for publication or to continue development or running test cases on their research programs. This was exactly the kind of counterproductive bureaucracy we had struggled to avoid in the early phases of the SUMEX-AIM resource, but were increasingly forced into because of pressures from BRTP.
- As we moved to a distributed model of computing, it was becoming increasingly difficult to assess realistically a measure of resource usage on which to base fee-for-service charges. Since many of the services used in such an environment do not have any accounting associated with them, because vendors do not provide the tools or because such measurements would add an unreasonable overhead to the basic service mechanism, we simply cannot allocate costs on the basis of usage because we do not have a valid measure of usage.

Thus, effective August 1, 1988, we discontinued the fee-for-service cost recovery system used during the precious two grant years. The operating cost shortfall is now being paid entirely by KSL core research projects (i.e., none of the Stanford or national collaborator projects pays anything for their increasingly communication-oriented use of SUMEX-AIM). The payments are negotiated between resource management and the KSL Principal

Investigators are implemented by direct charges against designated user research grants and contracts. This avoids the large accounting overhead of a cost center to collect relatively small bills each month.

III.E. Dissemination of Resource Information

We have continued our past substantial efforts to disseminate the AI technology developed at SUMEX-AIM. This has taken the form of many publications -- over forty-five combined books and papers are published per year by the KSL; wide distribution of our software, including systems software and AI application and tool software, both to other research laboratories and for commercial development; production of films and video tapes depicting aspects of our work; special seminars to introduce users to the systems we develop; and significant project efforts at studying the dissemination of individual applications systems such as the ONCOCIN resource-related research project. We continue to provide active support for the AIM workshops. The most recent one was held in the spring of 1988 at Stanford University, under the auspices of the American Association for Artificial Intelligence (AAAI). Planning is underway for an AIM workshop in the spring of 1990.

III.E.1. Software Distribution

We have widely distributed both our system software and our AI tool software. Since much of our general system-level software is distributed via the ARPANET we do not have complete records of the extent of the distribution. Software such as TOPS-20 monitor enhancements, the Ethernet gateway and TIP programs, the SEAGATE AppleNet to Ethernet gateway, the PUP Leaf server, the SUMACC development system for Macintosh workstations, and our Lisp workstation programs are frequently distributed in this manner to the ARPANET community and beyond.

Our primary distribution effort is directed towards the AI tools we have developed. In recent years, the volume of inquiries for this type of software and requests for tapes has been a substantial burden on the staff, especially in the face of recent budget cuts. As a result, we have turned over most of this type of software distribution to Stanford's Office of Technology Licensing (OTL). This organization handles software distribution and technology licensing matters for much of the Stanford community. Since there are several OTL staff members assigned to the distribution of Stanford software, requests for information and tapes are handled quickly and efficiently. Also, OTL's staff has the expertise needed to handle the legal questions that frequently arise in the distribution of software, and has an established computerized record-keeping scheme. The SUMEX staff continues to be available as needed to assist OTL with special administrative and technical matters.

Distribution continues for the Parallel Computing Architectures Project multiprocessor simulation system, CARE/SIMPLE, the EMYCIN package, and the BB1 package.

III.E.2. AIM Community Systems Support

We continue to make a special effort to assist other members of the SUMEX-AIM community in integrating the technologies needed for biomedical AI research. This is often achieved through direct contact by SUMEX staff members with researchers at these institutions, (e.g., with Professor Mark Frisse's and Michael Kahn's groups at Washington University, Dr. James Brinkley's group at the University of Washington, and Professor Widman's group at the University of Texas), at meetings and workshops, or via electronic mailing lists. For example, the Info-MAC, Info-Explorer, and Info-1100 mailing lists have hundreds of members and cover a broad range of equipment issues, software issues, and topics in artificial intelligence.

III.E.3. Video Tapes and Films

Various groups in the KSL have continued to prepare video tapes that provide an overview of the research and methodologies underlying our work and that demonstrate the capabilities of particular systems. These tapes are available through our groups, the Fleischmann Learning Center at the Stanford Medical Center, and the Stanford Computer Forum. In addition to the earlier tapes covering Knowledge Engineering in the KSL, ONCOCIN Overview, and ONCOCIN Demonstration, we have recent tapes on the PROTEAN project, the BB1 project, and a one-day symposium on KSL research activities.

III.E.4. Special Seminars

SIMPLE/CARE is a powerful simulation system which permits empirical studies of expert system performance on a wide class of multicomputer architectures, including quantitative measurements of system behavior. Our simulation system is now in use by several research groups at Stanford, and it has been ported to several external sites, including NASA Ames Research Center. A videotaped tutorial was held in June, 1988, attended by representatives from industry and government, which described the CARE/SIMPLE system, as well as the LAMINA programming interface. The attendees received instruction in use of the system for making measurements of the performance of various simulated multiprocessor applications.

Due to rapidly growing interest in the SIMPLE/CARE system, a major effort is now underway to port it to wider class of hardware platforms. The system is currently being reimplemented in Common Lisp and the X window system, with the Sun workstation as the initial target.

III.F. Suggestions and Comments

III.F.1. Resource Organization

We continue to believe that the Biomedical Research Technology Program is one of the most effective vehicles for developing and disseminating technological tools for biomedical research. The goals and methods of the program are well-designed to encourage building of the necessary multi-disciplinary groups and merging of the appropriate technological and medical disciplines.

III.F.2. Electronic Communications

SUMEX-AIM has pioneered in developing more effective methods for facilitating scientific communication. Whereas face-to-face contacts continue to play a key role, in the longer-term computer-based communications will become increasingly important to the NIH and the distributed resources of the biomedical community. We would like to see the BRTP take a more active role in promoting these tools within the NIH and its grantee community. This is particularly important in the light of significant on-going changes to the national networking environment.

IV. Description of Scientific Subprojects

The following subsections report on the AIM community of projects and "pilot" efforts, including local and national users of the SUMEX-AIM resource at Stanford. Many groups from the National AIM community now use the SUMEX-AIM resource solely for communication (i.e., electronic mail to and from colleagues or access to bulletin boards and other information resources at SUMEX). Because of the difficulty of recording Internet connections and system-level mail forwarding and related communications services, we can no longer accurately keep a list of these users. However, from the usage data shown in Section III.A.2.8, the volume of these services continues to rise as the AIM community moves increasingly to distributed resources.

The detailed collaborative project reports and comments are the result of a solicitation for contributions sent to each of the project Principal Investigators requesting the following information:

- I. Summary of Research Program
 - A. Project rationale
 - B. Medical relevance and collaboration
 - C. Highlights of research progress
 1. Accomplishments this past year
 2. Research in progress
 - D. List of relevant publications
 - E. Funding support
- II. Interactions with the SUMEX-AIM Resource
 - A. Medical collaborations and program dissemination via SUMEX
 - B. Sharing and interactions with other SUMEX-AIM projects (via computing facilities, workshops, personal contacts, etc.)
 - C. Critique of resource management (community facilitation, computer services, communications services, capacity, etc.)
- III. Research Plans
 - A. Project goals and plans
 1. Near-term
 2. Long-range
 - B. Justification and requirements for continued SUMEX use
 - C. Needs and plans for other computing resources beyond SUMEX-AIM
 - D. Recommendations for future community and resource development

We believe that the reports of the individual projects speak for themselves as rationales for participation. In any case, the reports are recorded as submitted and are the responsibility of the indicated project leaders. The only exceptions are the respective lists of relevant publications which have been uniformly formatted for parallel reporting on the Scientific Subproject Form.

IV.A. Stanford Projects

The following group of projects is formally approved for access to the Stanford aliquot of the SUMEX-AIM resource. Their access is based on review by the Stanford Advisory Group and approval by Professor Shortliffe as Principal Investigator.

IV.A.1. Guardian Project

Project Leader: Barbara Hayes-Roth, Ph.D.
Department of Computer Science
Stanford University

Collaborator: Adam Seiver, M.D.
Department of Surgery
Veterans Administration Hospital
Palo Alto, CA.

I. Summary of Research Program

A. Project Rationale

Critical care depends upon sophisticated life-support technology. Devices such as the respirator, dialysis machine, and intra-aortic balloon pump maintain life until the patient's own organs heal and resume normal function. However, effective management of device-supported patients is complex, involving interpretation of a large number of physiological variables, comparative evaluation of multiple therapeutic options, and control of many device parameters. Even skilled clinicians can make errors that produce life-threatening situations or otherwise harm the patient. These problems are compounded when the number of patients requiring life-support technology exceeds the availability of skilled clinicians.

Short-term research on computer-based assistance for critical care aims to alleviate some of these problems. Research on "smart alarms" aims to improve capabilities for signaling abnormal patient data, while reducing the false alarm rates associated with current alarm systems. Research on automatic tracking and combining of patient data values aims to help clinicians identify a wider range of problems. Research on display technology aims to help clinicians select, examine, and interpret important patient data. However, these short-term approaches do not address the fundamental problem of effectively managing an increasingly complex and sophisticated life-support technology.

From a longer-term perspective, what is needed is an "intelligent" computer system that integrates knowledge of underlying causal mechanisms and knowledge of the broader patient context--knowledge that currently is distributed among different experts on the critical care team. Such a system would acquire patient data automatically, synthesize data into a dynamic model of the patient's physiological functioning, and dynamically plan effective programs of device settings and other therapeutic actions. Acting in the role of an intelligent critical-care consultant, it would explain its observations, reasoning, conclusions, and recommendations to clinical care staff. Working toward this longer-term objective, we are developing "Guardian," a prototype system for intelligent patient monitoring in the surgical intensive care unit (SICU).

Development of Guardian poses many challenging research problems, including the following. How should we represent structure/function knowledge of biological systems and life-support devices? How should we represent the time-varying course of component biological processes and of the patient's medical condition? What kind of architecture will allow Guardian to integrate signal processing, knowledge-based reasoning, and user interaction? How can Guardian monitor a growing number of patient data parameters, many of them sampled several times per second, with limited computational resources? How should Guardian perform each of its component reasoning tasks? How can Guardian control its reasoning behavior--choice of reasoning tasks and strategies for performing particular tasks--so as to meet real-time constraints on the utility of its conclusions? These questions provide foci for the research of individual students on the project.

B. Medical Relevance and Collaboration

The proposed research aims to bring about a fundamental advance in the medical device technology base. Life-support devices are an accepted foundational element of critical care and they continue to advance in power and sophistication. However, like much advanced equipment, increasingly sophisticated life-support devices contribute to an increasingly complex information processing task for the people who use them. In the best of circumstances, cognitive overload threatens to undermine the utility of these devices and the quality of critical care. Short-term research directed at smart alarms, low-level data interpretation, and effective data displays will produce useful products, but it will not solve this long-term problem.

The proposed research attacks the long-term problem of effective critical care management with the methods of artificial intelligence. In essence, we aim to capture the knowledge and skills of critical care experts in a compute program that could assist skilled clinicians or stand in for unavailable staff. Thus, our goal is to create a device-management technology that is every bit as powerful as the device technology itself. We believe that this is the only way to fully exploit evolving life-support technology and insure high quality patient care.

In particular, an intelligent critical care consultant potentially could provide three kinds of assistance:

First, it could enhance the management and care of severely injured persons. It could guarantee the availability of expertise held by different medical experts (e.g., surgeon, nurse, physician specialists) at the time it is needed. It could provide continuous and vigilant patient monitoring. It could help attending clinicians to notice easily overlooked events, to analyze problems in sufficient detail, to consider alternative interpretations and treatments, and to avoid making errors.

Second, it could reduce workload and facilitate care tasks for providers. It could perform routine patient monitoring and provide summary accounts for

analysis and decision-making by clinicians. It could substitute for physician specialists whose expertise is required elsewhere. It could assume responsibility for managing stable patients, freeing available clinicians to focus on more seriously injured patients.

Third, it could provide continuing training and consultations to students or less well trained critical care staff.

Because the Guardian project depends upon state-of-the-art artificial intelligence methods, as well as extensive and sophisticated medical knowledge, it is being conducted as a collaboration between Dr. Barbara Hayes-Roth and Dr. Adam Seiver. Dr. Hayes-Roth designed the BB1 dynamic control architecture, which is the foundation for Guardian and several other applications involving real-time monitoring and control in other domains. Dr. Seiver is Associate Director of the new 14 bed surgical intensive care unit at the Palo Alto Veterans Administration Medical Center. At this time, the project also includes two post-doctoral fellows, Dr. Rattikorn Hewett and Dr. Luc Boureau, as well as three Ph.D. students, Mr. Richard Washington, Mr. Adnan Darwiche, and Mr. David Ash. In addition, we cooperate informally with Dr. Lawrence Fagan and his students, who are involved in the VentPlan project, which takes a control theoretic approach to complementary aspects of SICU monitoring, primarily the moment-by-moment optimization of device settings.

C. Research Progress

Following a period of task analysis, knowledge acquisition, and conceptual design, we developed the first version of Guardian in the winter of 1988. Since then, we have iterated a series of research cycles involving: (a) conceptual development of component approaches to knowledge representation and particular reasoning tasks; (b) implementation and integration of component approaches in a new version of Guardian; (c) demonstration of the new version on an important SICU scenario; and (d) solicitation of feedback and criticism from knowledgeable colleagues. Each such cycle takes about two academic quarters.

The current version of Guardian integrates these reasoning components: FOCUS preprocesses — abstracts, classifies, filters, and rates for urgency and importance — sensed data before relaying them to the reasoning system. FOCUS currently performs these functions for about 25 continuously monitored patient-data parameters. ASSOC-REACT uses probabilistic associations to diagnose commonly occurring problems and to suggest standard treatments. Recent innovations involving hierarchical organization of knowledge permit ASSOC-REACT to circumvent the well-known computational complexity of its underlying belief network mechanism and, in fact, to modulate its processing time according to the time available. ICE uses explicit structure/function models of biological systems — currently the respiratory, circulatory, and tissue metabolism systems — to suggest plausible underlying causes for unfamiliar or otherwise inexplicable

conditions. Recent innovations permit ICE to predict the consequences of observed or hypothetical conditions as well. TPLAN suggests courses of therapy actions over time to deal with evolving medical conditions. Each of these reasoning components is implemented in a domain-independent fashion and can be applied to any biological or other system for which the relevant knowledge is available.

D. Demonstrations, Presentations, and Publications

We have given demonstrations of the Guardian system to many colleagues in the medical AI and larger AI communities, for example to: Dr. Lawrence Fagan and his students from Stanford's Medical Computing Systems Group; Dr. Seppo Kalli, Director of Medical Signal Processing at Technical Research Centre of Finland; Dr. William Pardee and his associates from the Rockwell Science Center; Dr. Perry Thorndyke and his associates from FMC Corporation; Dr. Joseph Naser of the Electric Power Research Institute.

We have made (or plan to make) invited presentations of this research to: IEE Workshop on Expert Control Systems, Brighton England, June, 1990 International Joint Conference on Artificial Intelligence, Detroit, Aug, 1989 AI Systems in Government Conference, Washington D.C., March, 1989 AAAI Symposium on Knowledge System Development Tools, Stanford, March, 1989 Stanford SIGLunch, February, 1989 Workshop on Formal Aspects of Semantic Networks, Catalina, February, 1989 Carnegie Symposium on Architectures for Intelligence, Pittsburgh, May, 1988 Advanced Decision Systems, Palo Alto, May, 1988 Boeing Computer Services, Bellevue, WA., March, 1988 DARPA Knowledge-Based Planning Workshop, Austin, December, 1987

The project has produced the following publications:

- Hayes-Roth, B., Washington, R., Hewett, R., Hewett, M., and Seiver, A. Intelligent monitoring and control. Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-89, Detroit, Mi., 1989.
- Washington, R., and Hayes-Roth, B. Input data management in real time AI systems. Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-89, Detroit, Mi., 1989.
- Hayes-Roth, B. Making intelligent systems adaptive. In K. VanLehn (Ed.) Architectures for Intelligence. Lawrence Erlbaum, 1989.
- Hayes-Roth, B., Hewett, M., Washington, R., Hewett, R., and Seiver, A. Distributing intelligence within a single individual. In L. Gasser and M.N. Huhns (Eds.) Distributed Artificial Intelligence Volume 2. Morgan Kaufmann, 1989.
- Hewett, R., and Hayes-Roth, R. Representing and reasoning about physical systems using generic models. In J. Sowa (Ed.) Formal Aspects of Semantic Networks. Morgan Kaufmann, 1989.

- Hayes-Roth, B. Dynamic control planning in adaptive intelligent systems. Proceedings of the DARPA Knowledge-Based Planning Workshop, 1989.

IV.A.2. MOLGEN Project

MOLGEN - Applications of Artificial Intelligence to Molecular Biology: Research in Theory Formation, Testing, and Modification

Prof. E. Feigenbaum
Department of Computer Science
Stanford University

Dr. P. Friedland
NASA-Ames Research Center
Moffett Field, CA

Prof. Charles Yanofsky
Department of Biology
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

The MOLGEN project has focused on research into the applications of symbolic computation and inference to the field of molecular biology. This research has taken the specific form of systems that provide assistance to the experimental biologist in various tasks, including the design of complex experiment plans, the analysis of nucleic acid sequences, and the formulation of hypotheses in the subdomain of regulatory genetics.

B. Medical Relevance and Collaboration

The field of molecular biology has reached the point where the results of current research have immediate and important application to the pharmaceutical and chemical industries. Clinical testing has begun with synthetic interferon and human growth hormone produced by recombinant DNA technology. Governmental reports estimate that there are more than two hundred new and established industrial firms already undertaking product development using these new genetic tools.

The programs being developed in the MOLGEN project have already proven useful and important to a considerable number of molecular biologists. Currently several dozen researchers in various laboratories at Stanford (Prof. Paul Berg's, Prof. Stanley Cohen's, Prof. Laurence Kedes', Prof. Douglas Brutlag's, Prof. Henry Kaplan's, and Prof. Douglas Wallace's) and over four hundred others throughout the country have used MOLGEN programs over the SUMEX-AIM facility. We have exported some of our programs to users outside the range of our computer network (University of Geneva [Switzerland], Imperial Cancer Research Fund [England], and European Molecular Biology Institute [Heidelberg] are examples). The pioneering work on SUMEX has led to the establishment of a separate NIH-supported facility, BIONET, to serve the academic molecular biology research community with MOLGEN-like software. BIONET is now serving many of the computational

needs of over two thousand academic molecular biologists in the United States.

Our recent work in using qualitative reasoning techniques to encode theories of molecular biology is likely to be relevant to the human genome project. We have constructed models of the structure and function of the tryptophan operon gene-regulation system, including its enzymatic pathways, gene-regulation mechanisms, and the general processes involved in gene expression such as transcription and translation. The methods we developed to model this bacterial system will be applicable to modeling both the regulation of the thousands of genes that will be sequenced in the human genome project, and the activities of the protein products of these genes. Our work in hypothesis formation will aid molecular biologists in formulating to explain such processes as gene regulation, by referring to such large knowledge bases of biological knowledge.

C. Highlights of Research Progress

C.1 Accomplishments

The MOLGEN project has successfully concluded with the publication of Peter Karp's doctoral dissertation. Here we summarize the contributions of that dissertation.

Karp's dissertation investigates scientific reasoning from a computational perspective. The investigation focuses upon a program of research in molecular biology that culminated in the discovery of a new mechanism of bacterial gene regulation, namely Dr. Charles Yanofsky's discovery of attenuation. In the first phase of this research, the MOLGEN group performed a historical study of the biological research that reconstructed the different theories that the biologists possessed at different points in time, and analyzed the differences between successive theories. In the second phase, Karp developed a qualitative chemistry for representing theories of molecular biology. In the third phase of the research, he constructed a computer program that solves hypothesis-formation problems encountered by the biologists.

C.1.1 Qualitative Modeling and Simulation

In order to solve the hypothesis-formation task, we must have a framework for representing theories in molecular biology that allows those theories to be used to predict the outcomes of laboratory experiments. The representation must also allow the hypothesis-formation program to reason about a theory and modify it in order to improve the predictive power of the theory. Chapter 3 of the dissertation presents three related methods for representing scientific theories. The third representation method was selected for use in conjunction with the hypothesis-formation task. This method breaks theories in molecular biology into several parts: A *class knowledge base* defines a taxonomic hierarchy of the classes of biological objects that exist in the trp operon. A *process knowledge base* describes the chemical reactions that can

occur between biological objects. An experiment is described in a third knowledge base by creating the particular objects (instantiated from the known classes of objects) that are present in the experiment.

A qualitative reasoning program called GENSIM determines what reactions occur between the objects in an experiment; these reactions create new objects, which can cause additional reactions. Chapter 3 shows that a process-oriented framework for qualitative simulation is more flexible than a framework based on a fixed network of the state variables of a physical system. The GENSIM program defines a qualitative chemistry --- a framework for reasoning about chemical reactions --- and identifies constraints that a chemical modeling system must satisfy to simulate chemical reactions correctly. In addition, the chapter presents new qualitative representations for capturing the partial knowledge that biologists (and other scientists) have of the mathematical relationships that characterize the systems they study.

C.1.2 A Historical Study of the Discovery of Attenuation

Chapter 4 contains a detailed historical study of the process by which Dr. Yanofsky and his colleagues discovered attenuation. The study is based on information obtained from the scientific publications the biologists produced, and from interviews with the biologists. This biological research program consumed over 50 man-years of effort, and so is one of the most complex ever studied in Artificial Intelligence.

In the first phase of the analysis, the MOLGEN group produced a conceptual reconstruction of what knowledge the biologists possessed about the *trp* operon at different points in time. In the next phase, Karp searched for patterns in the differences between successive states of the biologists' knowledge. These differences were due to changes the biologists made to their theories of the *trp* operon. Patterns in the differences indicate reasoning methods that were used to derive new theories from old. This analysis suggests that biologists use *theory modification operators* to modify a theory such that its predictions are altered. These patterns also support the conjecture that scientists use four different *modes of scientific exploration* to determine what types of experiments to perform next from a given state of research. A mode of exploration selects experiments based on the number of theories entertained at a given moment, and their relative credibilities.

C.1.3 Hypothesis Formation

Chapter 5 describes methods for solving the hypothesis-formation problem. The problem is to generate hypotheses that rectify a discrepancy between the observed outcome of an experiment, and the outcome predicted by GENSIM. A hypothesis modifies either GENSIM's theory or the initial conditions of the experiment (which are often not known with certainty), such that the predicted outcome of the experiment matches its observed outcome. The thesis treats the problem of hypothesis formation as a *design problem*. The goal of the designer is to eliminate the difference between the observed and

predicted outcomes of the experiment -- the *prediction error*. A hypothesis is synthesized by *design operators* that reason backward from the prediction error and determine what modifications to the theory or initial experimental conditions will eliminate the error. HYPGENE uses a sophisticated planning system to perform backward reasoning; the planner can achieve goals represented as arbitrary predicate-calculus formulae, including quantification.

This design problem is a search problem because often more than one operator is relevant to eliminating a prediction error, and a single operator can sometimes be applied in several ways. The synthetic, goal-directed search used here should prove more efficient than past approaches to hypothesis generation, which often used heuristic search to guide a purely syntactic generator of hypotheses. HYPGENE uses heuristic search to guide a generator that is already focused on errors in the prediction. Its search can be guided further by the results of a second, similar experiment which we term a *reference experiment*. The difference between the initial conditions of the two experiments is likely to have caused the prediction error, and can be used evaluate hypotheses HYPGENE has generated. In addition, the thesis describes a reasoning mode which can generate hypotheses by reasoning forward from the difference in initial conditions. This forward reasoning mode is more efficient than the backward mode for some problems.

Chapter 7 is an empirical investigation of the methods in Chapters 3 and 5, in which GENSIM and HYPGENE were run on several test cases. GENSIM predicted the outcomes of several biological experiments. HYPGENE formulated hypotheses to account for several incorrect GENSIM predictions. Most of these hypothesis-formation problems were taken from the historical study of the biologists research; HYPGENE produced many of the same hypotheses as the biologists did. The chapter summarizes the strengths and weaknesses of the methods, as revealed by these tests.

HYPGENE provides a *flexible* framework for hypothesis formation because its framework is syntactically complete -- its operators can modify the initial conditions of the experiment, the process knowledge base, or the class knowledge base. HYPGENE's flexibility is also enhanced because its planner can manipulate complex predicate-calculus expressions, allowing the program to reason about complex domain processes. Because HYPGENE's planner and operators do not contain domain concepts, the framework is largely domain independent. The framework is efficient because HYPGENE's planner works backward from prediction errors using operators that associate syntactic classes of prediction errors with specific types of theory modifications. The framework allows us to integrate domain-specific knowledge (such as general knowledge of chemistry) into the hypothesis generator to prune partial solutions during the generation process. Efficiency is further increased by the use of reference experiments, which provide information for both filtering and generating hypotheses.

D. Publications

1. Bach, R., Friedland, P., and Iwasaki, Y.: *Intelligent computational assistance for experiment design*. Nucleic Acids Res. 12(1):11-29, January, 1984.
2. Friedland, P., and Kedes, L.: *Discovering the secrets of DNA*. Communications of the ACM, 28(11):1164-1186, November, 1985, and IEEE/Computer, 18(11):49:69, November, 1985.
3. Friedland, P. and Iwasaki Y.: *The concept and implementation of skeletal plans*. Journal of Automated Reasoning, 1(2): 161-208, 1985.
4. Friedland, P., Armstrong, P., and Kehler, T.: *The role of computers in biotechnology*. BIO\TECHNOLOGY 565-575, September, 1983.
5. Karp, P., and D. Wilkins: *An Analysis of the Deep / Shallow Distinction for Expert Systems*. To be published in International Journal of Expert Systems, 1988.
6. Karp, P., and P. Friedland: *Coordinating the Use of Qualitative and Quantitative Knowledge in Declarative Device Modeling*, in Artificial Intelligence, Simulation, and Modeling edited by L. Widman and K. Loparo, 1989.
7. Karp, P.: *A Process-Oriented Model of Bacterial Gene Regulation*, Stanford University Knowledge Systems Laboratory Technical Report KSL-88-18.
8. Round, A.: *QSOPS: A Workbench Environment for the Qualitative Simulation of Physical Processes*. Stanford University Knowledge Systems Laboratory Report KSL-87-37, 1987.
9. Karp, P.: *Hypothesis Formation by Design*, in Computational Models of Scientific Theory Formation, edited by J. Shrager and P. Langley, 1989, in press.
10. Karp, P.: *Hypothesis Formation and Qualitative Reasoning in Molecular Biology*, PhD Dissertation, Stanford University Computer Science Department, 1989.
11. Meyers, S. and Friedland, P.: *Knowledge-based simulation of regulatory genetics in bacteriophage Lambda*. Nucleic Acids Res. 12(1):1-9, January, 1984.

E. Funding Support

The MOLGEN grant, which has supported the bulk of this research, is titled: MOLGEN: Applications of Artificial Intelligence to Molecular Biology: Research in Theory Formation, Testing, and Modification. This NSF Grant number MCS-8310236, expired on 10/31/86. The Principal Investigators were Edward A. Feigenbaum, Professor of Computer Science and Charles Yanofsky, Professor of Biology. Additional support for this research was

provided by the Defense Advanced Research Projects Agency, under contract N00039-86C-0033.

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

SUMEX-AIM continued to serve as the nucleus of our computing resources. The facility not only provided excellent support for our programming efforts, but served as a major communication link among members of the project. Systems available on SUMEX-AIM such as EMACS, MM, Scribe and BULLETIN BOARD have made possible the project's documentation and communication efforts.

We have taken advantage of the collective expertise on medically-oriented knowledge-based systems of the other SUMEX-AIM projects. In addition to especially close ties with other projects at Stanford, we have greatly benefited from interaction with other projects at yearly meetings and through exchange of working papers and ideas over the system.

III. RESEARCH PLANS

This project has concluded successfully with the completion of Dr. Karp's thesis; no future research is planned at this time.

IV.A.3. ONCOCIN Project

Principal Investigator: Edward H. Shortliffe, M.D., Ph.D.
Departments of Medicine and Computer Science
Stanford University

Project Director: Lawrence M. Fagan, M.D., Ph.D.
Department of Medicine
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

The ONCOCIN Project is one of many Stanford research programs devoted to the development of knowledge-based expert systems for application to medicine and the allied sciences. The central issue in this work has been to develop a program that can provide advice similar in quality to that given by human experts, and to ensure that the system is easy to use and acceptable to physicians. The work seeks to improve the interactive process, both for the developer of a knowledge-based system, and for the intended end user. In addition, we have emphasized clinical implementation of the developing tool so that we can ascertain the effectiveness of the program's interactive capabilities when it is used by physicians who are caring for patients and are uninvolved in the computer-based research activity.

B. Medical Relevance and Collaboration

The lessons learned in building prior production rule systems have allowed us to create a large oncology protocol management system much more rapidly than was the case when we started to build MYCIN. We introduced ONCOCIN for use by Stanford oncologists in May 1981. This would not have been possible without the active collaboration of Stanford oncologists who helped with the construction of the knowledge base and also kept project computer scientists aware of the psychological and logistical issues related to the operation of a busy outpatient clinic.

C. Highlights of Research Progress

C.1.A Background and Overview of Accomplishments

The ONCOCIN Project is a large interdisciplinary effort that has involved over 35 individuals since the project's inception in July 1979. The work is currently in its tenth year; we summarize here the milestones that have occurred in the research to date:

The first version was a character-based system run on the large DEC mainframes. Because of the limitations in the display capabilities and the requirement to be tethered to a time-shared mainframe computer, we began a re-implementation of the system in 1984. In 1985, we discontinued the mainframe version of ONCOCIN. The performance of the mainframe version

of ONCOCIN was documented in two evaluation papers that appeared in clinical journals (see Hickam and Kent's papers).

In 1986, we placed the workstation version of ONCOCIN into the Oncology Day Care clinic. This version is a completely different program from the version of ONCOCIN that ran on the DECsystem 20--using protocols entered through the OPAL program, with a new graphical data entry interface, and a revised knowledge representation and reasoning component.

The process of entering a large number of treatment protocols in a short period of time led to other research topics including: design of an automated system for producing meaningful test cases for each knowledge base, modification of the design and access methods for the time-oriented database, and the development of methods for graphically viewing multiple protocols that are combined into one large knowledge base.

In 1987, we began to explore the use of continuous speech recognition as an alternate entry method for communicating with ONCOCIN. This project requires the connection of speech recognition equipment produced by Speech Systems, Inc. of Tarzana to the ONCOCIN interface module. Christopher Lane has developed a prototype network connection and command interpreter between the speech module and the Xerox 1186 computer that runs ONCOCIN. Clifford Wulfman has designed a series of modifications to the ONCOCIN user interface to allow for verbal commands. This work is described in more detail in the core ONCOCIN section.

The majority of our effort during the last two years has been to understand the limitations of the clinic version of ONCOCIN, and to concentrate on the generalization of these techniques to other application areas besides oncology. The majority of this research is thus described as part of the core research discussion on ONCOCIN. Highlights of this year include: (1) development of a general knowledge acquisition tool (PROTEGE) designed to handle skeletal planning applications for clinical trials in any area of medicine, (2) demonstration that the therapy planning and knowledge acquisition tools for ONCOCIN can be closely integrated, and (3) development of a speech input system for ONCOCIN.

As a demonstration of the capabilities of the project to date, we undertook an experiment to see how difficult and time-consuming it is to bring up a new treatment protocol. A summary of a recent colon protocol was down-loaded from the PDQ protocol database. Approximately 60% of the knowledge of the protocol summary fit easily into the OPAL high level description. Additional rules were entered using lower level editors. A limited consultation was run after about 4 hours of work. Although this is only one data point, we believe that it validates the generality of the knowledge acquisition and therapy planning approach that we have pursued for nearly a decade. Work continues on extending the knowledge acquisition and therapy planning tools to allow for a higher percentage of concepts that can be entered with the smallest possible amount of low level Lisp changes.

Although we have completed the transfer of ONCOCIN into a stable and useful system on the Xerox Lisp workstations, it is now clear that this type of machine will not provide the type of dissemination hardware we would like to see. There are no planned additions to increase the speed, decrease the cost, or increase the integration capabilities of these workstations. Although there may be other solutions that will allow us to port ONCOCIN directly to alternative hardware platforms, we need to move away from Xerox workstations and InterLisp language upon which most of our software is based. We are particularly interested in exploring the Mac II hardware. During the last six months, we have begun an experiment to port ONCOCIN to a TI Explorer board inside of a Mac II. We have completed the translation of the Ozone object-oriented system, the temporal network and most of the reasoner. We will next approach the design of the user interface, which must be rewritten anew, since the current interface depends heavily on the graphical capabilities of the Xerox workstations.

C.1.B Review of Research Issues in ONCOCIN and OPAL

Our work to refine the clinic versions of ONCOCIN and OPAL reached a mature stage during this last research year. As our attention has moved to the generalization of these tasks (E-ONCOCIN and PROTÉGÉ) it seems appropriate to describe the range of research issues that we have examined during the development of the ONCOCIN system.

Research Issues in the Development of the ONCOCIN Reasoner and Interviewer

- *Redesign of the reasoning component.* A major impetus for the redesign of the system was to develop more efficient methods to search the knowledge base during the running of a case. We have implemented a reasoning program that uses a discrimination network to process the cancer protocols. This network provides for a compact representation of information which is common to many protocols but does not require the program to consider and then disregard information related to protocols that are irrelevant to a particular patient. We continue to improve portions of the reasoning component that are associated with reasoning over time; e.g., modeling the appropriate timing for ordering tests and identifying the information which needs to be gathered before the next clinic visit. In general, we are concentrating on improving the representation of the knowledge regarding sequences of therapy actions specified by the protocol.

Our experience with adding a large number of protocols has led to the evaluation of the design of the internal structure of the knowledge base (e.g., the way we describe the relationships between chemotherapies, drugs, and treatment visits). We will continue to improve the method for traversing the plan structure in the knowledge base, and consider alternative arrangements for representing the structure of chemotherapy plans. Currently, the knowledge base of treatment guidelines and the

patient database are separated. We propose to tie these two structures closer together. Additional work is anticipated on turning ONCOCIN into a critiquing system, where the physician enters their therapy and ONCOCIN provides suggestions about possible alternatives to the entered therapy. Although we have concentrated our review of the ONCOCIN design primarily on the data provided by additional protocols, we know that non-cancer therapy problems may also raise similar issues. The E-ONCOCIN effort is designed to produce a domain-independent therapy planning system that includes the lessons learned from our oncology research. Samson Tu is primarily responsible for continued improvement of the reasoning component of ONCOCIN.

- *Extensions to the user interface.* We continue to experiment with various configurations of the user interface. Many of the changes have been in response to requests for a more flexible data management environment. We are occasionally faced with data that becomes available corresponding to a time before the current visit. This can happen if a laboratory result is delayed, or a patient's electronic flowsheet is started in the middle of the treatment. We have added the ability to create new columns of data, and are designing the changes to the temporal processing components of ONCOCIN to allow for data that is inserted out of order. We have also extended the flowsheet to allow for patient specific parameters (e.g., special test results or symptoms) that the physician wishes to follow over time. The flowsheet layouts have been modified to create protocol specific flowsheets, e.g., lymphoma flowsheets have a different configuration than lung cancer flowsheets. The basic structure of the interface has been modified to use object-oriented methods, which allows for more flexible interaction between different components of the flowsheet and the operations performed on the flowsheet.

A continuing area of research concerns how to guide the user to the most appropriate items to enter (based on the needs of the reasoning program) without disrupting the fixed layout of the flowsheet. The mainframe version of ONCOCIN modified the order of items on the flowsheet to extract necessary information from the user. In the workstation version, we have developed a guidance mechanism which alerts the user to items that are needed by the reasoning program. The user is not required to deviate from a preferred order of entry nor required to respond to a question for which no current answer is available. Cliff Wulfman is primarily responsible for improvements to the user interface of ONCOCIN.

- *System support for the reorganization.* The LISP language, which we used to build the first version of ONCOCIN, does not explicitly support basic knowledge manipulation techniques (such as message passing, inheritance techniques, or other object-oriented programming structures). These facilities are available in some commercial products, but none of the existing commercial implementations provide the reliability, speed,

size, or special memory-manipulation techniques that are needed for our project. We have therefore developed a "minimal" object-oriented system to meet our specifications. The object system is currently in use by each component of the new version of ONCOCIN and in the software used to connect these components. In addition, all ONCOCIN student projects are now based on this programming environment. Christopher Lane created and is responsible for modifications to the object-oriented system.

Interactive Entry of Chemotherapy Protocols by Oncologists (OPAL)

The OPAL system permits physicians who are not computer programmers to enter protocol information on a structured set of forms presented on a graphics display. Most expert systems require tedious entry of the system's knowledge. In many other medical expert systems, each segment of knowledge is transferred from the physician to the programmer, who then enters the knowledge into the expert system. We have taken advantage of the generally well-structured nature of cancer treatment plans to design a knowledge entry program that can be used directly by clinicians. The structure of cancer treatment plans includes:

- choosing among multiple protocols (that may be related to each other);
- describing experimental research arms in each protocol;
- specifying individual drugs and drug combinations;
- setting the drug dosage level;
- and modifying either the choice of drugs or their dosage.

Using the graphics-oriented workstations, this information is presented to the user as computer-generated forms which appear on the screen. After the user fills in the blanks on the forms, the program generates the rules used to drive the reasoning process. As the user describes more detailed aspects of the protocol, new forms are added to the computer display; these allow the user to specify the special cases that make the protocols so complicated. Although the user is unaware of the creation of the knowledge base from the interaction with OPAL, a complex set of translations are taking place. The user's entries are mapped into an intermediate data structure (IDS) that is common for all protocols. From the IDS, a translation program generates rules for creating and modifying treatment, and integrates them with the existing ONCOCIN knowledge base. Considerable effort has been expended on producing a standard relational database as the appropriate data structure to underlie the OPAL IDS. The PROTÉGÉ system described in the core ONCOCIN section was built upon this relational database.

Although the "forms" were specifically designed for cancer treatment plans, the techniques used to organize data can be extended to other clinical trials, and eventually to other structured decision tasks. The key factor is to exploit the regularities in the structure of the task (e.g., this interface has an extensive notion of how chemotherapy regimens are constructed) rather than to try to build a knowledge-entry system that can accept *any* possible problem

specification. The OPAL program is based upon a domain-independent forms creation package designed and implemented by David Combs. This program will provide the basis for our extension of OPAL to other application areas.

We have now entered thirty-five protocols covering many different organ systems and styles of protocol design. Based on this experience, we continue to explore ways to modify OPAL to increase the percentage of the protocol that can be entered directly by our clinical collaborators. One direction in which we have extended the OPAL program is in providing a graphical interface of nodes and arcs to specify the procedural knowledge about the order of treatments and important decision points within the treatments. This work is described in several papers by Musen.

C.2 Research in Progress

The major thrusts in speech input and generalized knowledge acquisition are described in the core research description of ONCOCIN. We will describe here our research in complex therapy planning and its spin-offs in temporal representations and summarization of patient records.

C.2.1 Strategic Therapy Planning (ONYX)

We have continued our research project (ONYX) to study the therapy-planning process and to determine how clinical strategies are used to plan therapy in unusual situations. Our goals for ONYX are: (1) to conduct basic research into the possible representations of the therapy-planning process, (2) to develop a computer program to represent this process, and (3) eventually to interface the planning program with ONCOCIN. We have worked with our clinical collaborators to determine how to create therapy plans for patients whose special clinical situation preclude following the standard therapeutic plan described in the protocol document.

The prototype program design has four components: (1) to review the patient's past record and recognize emerging problems, (2) to formulate a small number of revised therapy plans based on existing problems, (3) to determine the results of the generated plans by using simulation, and (4) to weight the results of the simulation and rank order the plans by performing decision analysis. This model is described in the papers by Langlotz.

We have built an expert system based on decision analytic techniques as part of the solution to the fourth step of the ONYX planning problem. The program carries out a dialogue with the user concerning the particular treatment choices to be compared, potential problems with the treatments, and the patient-specific utilities corresponding to the possible outcomes. A decision tree is automatically created, displayed on the screen, and solved. The solution is presented to the user, and is compatible with an explanation program for decision trees being developed as part of the Ph.D. research of Curtis Langlotz.

A major spin-off from our ONYX work is a program that can summarize temporal trends in patient visits during chemotherapy and produce a

summary of the patient's course using both data from the flowsheet and an *underlying model of bone marrow physiology*. This work has led to major improvements in the temporal representation and in the integrate of mathematical and symbolic models. This work is part of Michael Kahn's Ph.D. thesis.

Summarization is defined as the task of combining multiple observations or features into a more general statement and abstraction as the task of selecting a subset of available features considered most relevant to answering a particular question. Both tasks require a model of the underlying system that encodes extensive knowledge about the entities and relationships that cause the system behavior and result in the observations. In the setting of a dynamic system, the model must be capable of representing temporal relationships between entities.

This work proposes that the combination of mathematical and symbolic techniques can be used to construct useful summaries of complex time-ordered data. In particular, mathematical models are used to capture the knowledge about the physiological processes that are responsible for the patient's clinical findings. Model parameters represent physiological concepts that are clinically relevant for medical problem solving. Prior to any patient-specific observations, the model parameters are set to population-based estimates. Standard curve-fitting techniques using a Bayesian updating scheme adjust model parameters to new observations. As more patient-specific observations are obtained, the set of estimated model parameters move further away from the population estimates. Symbolic models are used to augment the mathematical model parameter and state estimates. As the patient's clinical course evolves, the symbolic model captures the concurrent contexts that affect the interpretation of the physiological model results. For example, a heart rate of 120 is considered abnormally high in the context of a resting person but may be inappropriately low in the context of a treadmill stress test. A key feature of the combined mathematical and symbolic approach is that the physiological model changes over time as additional data are obtained and the symbolic model modifies the interpretation of these model changes in light of the clinical contexts present when the data was observed.

The methodology for combining mathematical and symbolic models emphasizes four main elements in summarizing complex time-ordered data:

- 1) A mechanistically-motivated model (in medicine, a physiological model) forms the basis for converting raw observations into more meaningful concepts. However, the interpretation of these concepts requires additional knowledge such as the contextual information contained in a symbolic model.
- 2) The initial model is based on general knowledge since no specific observations are available to alter the initial impression. New observations will change the initial model by incorporating the new

information. The collection of altered models captures state changes that have evolved over time.

- 3) Differences in key model features or states form the basis for selective abstraction and effective summarization. A method for determining which features are pertinent to a user question or sufficiently "interesting" to warrant inclusion into a summarization requires additional domain-specific reasoning.
- 4) The construction of a concise and useful summarization requires the use of additional contextual and domain-specific information so that the generated summary text conforms to the user's expectations and requirements.

These principles form the basis for a computer program designed to summarize the clinical course of individual patients receiving experimental cancer chemotherapy. In this setting, patients are often receiving more than one treatment that have overlapping schedules and durations of action. Thus our temporal model requires the representation and the reasoning with multiple, simultaneous contexts to ensure the proper interpretation of a given observation or model estimate. ONCOCIN uses a specialized structure called the temporal network to represent treatment contexts used in temporal queries into a time-oriented patient record. We have extended the temporal network concept to create a symbolic model of the patient's clinical course over time. This structure permits the representation of multiple, concurrent contexts over time and therefore can capture the complex temporal nature of our patient's clinical course. For the proper interpretation of the mathematical model output, the temporal network provides the set of contexts that existed when the observation and model estimates were obtained. In addition, the interpretation task requires complex context-sensitive reasoning. For example, the interpretation of a model parameter may be different if two contexts were present concurrently than if either context was present alone. The temporal network provides a mechanism for altering the available reasoning methods based on the set of current contexts. In this use of the temporal network, reasoning methods are associated with each context. When a context is present, a temporal network node representing that context is created and the reasoning methods are made available to the interpretation process. A temporal network node may also withdraw methods made available by other temporal network nodes. In this manner, a general rule or method can be suspended if it is not appropriate in particular context.

We believe that the combination of mathematical models along with specialized symbolic structures results in more representational and inferencing power than either method alone. Well established mathematical techniques convert observations into underlying system concepts while symbolic techniques interpret the mathematical results using additional domain knowledge. Although some of these features could possibly be represented using either mathematical or symbolic techniques alone, we

believe the resulting "pure" models would be too complex to be useful. In combining the two approaches, concepts best modeled in a physiological paradigm can be expressed within the mathematical model while concepts best modeled symbolically can be represented within the temporal network. We are continuing to explore this architecture in the VentPlan system designed to assist with data interpretation in the Intensive Care Unit application area.

C.2.2 Documentation

In 1986, we videotaped a lecture and demonstration of the ONCOCIN and OPAL systems at the XEROX Palo Alto Research Center. This videotape is available for loan from our offices. Our previous videotapes have been shown at scientific meetings and have been distributed to many researchers in other countries. The publications described below further document our recent work on ONCOCIN. We have sent copies of our system to the University of Pittsburgh, and have distributed it to the National Library of Medicine. We have developed a user manual, description of sample interaction, reference card, and graphical flowchart to help with training in the use of ONCOCIN.

D. Publications Since January, 1988

- 1) Musen, M.A. Generation of Model-Based Knowledge-Acquisition Tools for Clinical-Trial Advice Systems. KSL-88-06. Doctoral dissertation, Medical Information Sciences Program, Medical Computer Science Group, Stanford University, January 1988 Updated version appears as Automated Generation of Model-Based Knowledge-Acquisition Tools. London: Pitman, 1989.
- 2) Wulfman, C.E., Isaacs, E.A., Webber, B.L., and Fagan, L.M. Integration discontinuity: Interfacing users and systems. Memo KSL-88-12, February 1988. Proceedings of Architectures for Intelligent Interfaces: Elements and Prototypes, pp. 57-68, Monterey CA, March 29-April 1, 1988.
- 3) Musen, M.A. Conceptual models of interactive knowledge-acquisition tools. Report KSL-88-16, March 1988. Proceedings of European Knowledge Acquisition Workshop (EKAW'88), Bonn, FRG, June 1988. Gesellschaft für Mathematik und Datenverarbeitung (GMD) Technical Report #143. Knowledge Acquisition, in press.
- 4) Musen, M.A. Generation of knowledge-acquisition tools from clinical-trial models. Report KSL-88-26, March 1988. Lecture Notes in Medical Informatics (P.L. Reichertz and D.A.B. Lindberg, eds.), Vol. 35, pp. 630-635. Proceedings (Medical Informatics Europe '88), Oslo, Norway: Springer-Verlag, 17-20 August 1988.

- 5) Musen, M.A. An editor for the conceptual models of interactive knowledge-acquisition tools. Report KSL-88-44, June 1988. Proceedings of the Third Workshop on Knowledge Acquisition for Knowledge-Based Systems, Banff, Alberta, Canada, November 1988. Also to appear in the International Journal of Man-Machine Studies, in press. A shorter version of this paper was published in the Knowledge Acquisition Special Issue of SIGART Newsletter, 108:45-55, April 1989.
- 6) Musen, M.A. and van der Lei, J. Of brittleness and bottlenecks: Challenges in the creation of pattern-recognition and expert-system models. Report KSL-88-59, August 1988. Pattern Recognition and Artificial Intelligence: Towards an Integration of Techniques (E.S. Gelsema and L.N. Kanal, eds.), pp. 335-352. New York: Elsevier, 1988.
- 7) Tu, S.W., Kahn, M.G., Musen, M.A., Ferguson, J.C., Shortliffe, E.H. and Fagan, L.M. Episodic monitoring of time-oriented data for heuristic skeletal-plan refinement. Report KSL-87-70, August 1988. (submitted to CACM).
- 8) Langlotz, C.P. and Shortliffe, E.H. An analysis of categorical and quantitative methods for planning under uncertainty. Report KSL-88-63, September 1988. Proceedings of the Twelfth Annual Symposium on Computer Applications in Medical Care, pp. 114-118, Washington, D.C., November 1988.
- 9) Combs, D. The OPAL knowledge-acquisition tool: Functional specification. Technical documentation, internal working memo.
- 10) Combs, D. ODIE: A system for design and management of form-based interfaces. Technical documentation, internal working memo.
- 11) Shwe, M., Tu, S.W. and Fagan, L.M. Validating the knowledge base of a therapy-planning system. Report KSL-88-70. October 1988. Methods of Information in Medicine, 28(1):36-50, 1989. Musen, M.A. Generation of visual languages for development of knowledge-based systems. Report KSL-88-73, December 1988. To appear as a chapter in Visual Languages, Volume II (R.R. Korfhage, E. Jungert, and T. Ichikawa, eds.), New York: Plenum, 1989.
- 12) Shortliffe, E.H. Testing reality: The introduction of decision-support technologies for physicians. Report KSL-88-82, December 1988. Published as an editorial in Methods of Information in Medicine, 28(1):1-5, 1989.
- 13) Beinlich, I.A., Suermondt, H.J., Chavez, R.M., and Cooper, G.F. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. Report KSL-88-84, January 1989. To appear in AI in Medicine, London 1989.
- 14) Rutledge, G., Thomsen, G., Beinlich, I., Farr, B., Kahn, M., Sheiner, L., and Fagan, L. VentPlan: An architecture for combining qualitative and quantitative computation. Report KSL-89-04, January 1989.

- 15) Musen, M.A. Languages for knowledge acquisition: Building and extending models. Report KSL-89-07, January 1989. Proceedings of the AAAI Spring Symposium on Knowledge System Development Tools and Languages. American Association for Artificial Intelligence, Stanford CA, March 1989.
- 16) Musen, M.A. Widening the knowledge-acquisition bottleneck: Automated tools for building and extending clinical models. Report KSL-89-09, February 1989. Proceedings of the AAMSI Congress '89, 1989, pp 2-7.
- 17) Farr, B.R. Decision-theoretic evaluation of therapy plans. Report KSL-89-20. Submitted to the Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care, March 1989.
- 18) Reed, J. Building decision models that modify decision systems. Report KSL-89-21. Submitted to the Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care, March 1989.
- 19) Musen, M.A. Automated support for building and extending expert models. Report KSL-89-26, March 1989. To appear in a special issue of Machine Learning.
- 20) Musen, M.S. and van der Lei, J. Knowledge engineering for clinical consultation programs: Modeling the application area. Report KSL-89-28, March 1989. Methods of Information in Medicine, 28(1):28-35, 1989.
- 21) Kahn, M.G., Fagan, L.M., and Sheiner, L.B. Model-based interpretation of time-varying medical data. Report KSL-89-34, April 1989.
- 22) Langlotz, C.P. and Shortliffe, E.H. The critiquing approach to automated advice and explanation: Rationale and examples. KSL-89-46, May 1989. To appear in Expert Knowledge and Explanation: The Knowledge-Language Interface (C. Ellis, ed.), London: Ellis Horwood, 1989.

E. Funding Support

Grant Title: "Therapy-planning strategies for consultation by computer"

Principal Investigator: Edward H. Shortliffe

Project Management: Lawrence M. Fagan

Agency: National Library of Medicine

ID Number: LM-04136

Term: April 1987 to March 1990

Total award: \$380,123

Grant Title: Postdoctoral Training in Medical Information Science

Principal Investigator: Edward H. Shortliffe

Project Management: Edward H. Shortliffe

Agency: National Library of Medicine

ID Number: 1 T32 LM07033

Term: July 1, 1984 - June 30, 1989

Total award: \$903,718

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A. Medical Collaborations and Program Dissemination via SUMEX

A great deal of interest in ONCOCIN has been shown by the medical, computer science, and lay communities. We are frequently asked to demonstrate the program to Stanford visitors. We also demonstrated our developing workstation code in the Xerox exhibit in the trade show associated with AAAI-84 in Austin, Texas, IJCAI-85 in Los Angeles, AAAI-86 in Philadelphia, and Medinfo 86. Physicians have generally been enthusiastic about ONCOCIN's potential. The interest of the lay community is reflected in the frequent requests for magazine interviews and television coverage of the work. Articles about MYCIN and ONCOCIN have appeared in such diverse publications as *Time* and *Fortune*, and ONCOCIN has been featured on the "NBC Nightly News," the PBS "Health Notes" series, and "The MacNeil-Lehrer Report." Most recently it appeared in a special on Artificial Intelligence for TV Ontario (Canadian PBS station) and "Physician's Journal Update" on the Lifetime Cable Network. Due to the frequent requests for ONCOCIN demonstrations, we have produced a videotape about the ONCOCIN research which includes demonstrations ONCOCIN and OPAL. The tape has also been shown to both national and international researchers in biomedical computing. We are producing a tape that describes our speech research.

Our group also continues to oversee the MYCIN program (not an active research project since 1978) and the EMYCIN program. Both systems continue to be in demand as demonstrations of expert systems technology. MYCIN has been demonstrated via networks at both national and international meetings in the past, and several medical school and computer science teachers continue to use the program in their computer science or medical computing courses. Researchers who visit our laboratory often begin their introduction by experimenting with the MYCIN/EMYCIN systems. We also have made the MYCIN program available to researchers around the world who access SUMEX using the GUEST account. EMYCIN has been made available to interested researchers developing expert systems who access SUMEX via the CONSULT account. One such consultation system for psychopharmacological treatment of depression, called Blue-Box (developed by two French medical students, Benoit Mulsant and David Servan-Schreiber), was reported in July of 1983 in *Computers and Biomedical Research*. The EMYCIN experience is now well disseminated via commercial products.

B. Sharing and Interaction with Other SUMEX-AIM Projects

The community created on the SUMEX resource has other benefits which go beyond actual shared computing. Because we are able to experiment with other developing systems, such as INTERNIST/CADUCEUS, and because we frequently interact with other workers (at AIM Workshops or at other

meetings), many of us have found the scientific exchange and stimulation to be heightened. Several of us have visited workers at other sites, sometimes for extended periods, in order to pursue further issues which have arisen through SUMEX- or workshop-based interactions. In this regard, the ability to exchange messages with other workers, both on SUMEX and at other sites, has been crucial to rapid and efficient dissemination of ideas. Certainly it is unusual for a small community of researchers with similar scholarly interests to have at their disposal such powerful and efficient communication mechanisms, even among those researchers on opposite coasts of the country.

During this past three years, we have had extensive interactions with Randy Miller at Pittsburgh. Via floppy disks and SUMEX, we have experimented with several versions of the QMR program. The interaction was very much facilitated by the availability of SUMEX for communication and data transmission. Several recent papers have been written to describe collaborations between students in our training program and the group at the University of Pittsburgh and Carnegie-Mellon University.

C. Critique of Resource Management

Our community of researchers has been extremely fortunate to work on a facility that has continued to maintain the high standards that we have praised in the past. The staff members are always helpful and friendly, and work as diligently to please the SUMEX community as to please themselves. As a result, the computer is as accessible and easy-to-use as they can make it. More importantly, it is a reliable and convenient research tool. We extend special thanks to Tom Rindfleisch for maintaining such high professional standards. As our computing needs grow, we have increased our dependence on special SUMEX skills such as networking and communication protocols. As described above, we are moving our software development to a combination of Lisp processor boards and Mac II's, and will need increased networking capabilities in this environment.

III. RESEARCH PLANS

A. Project Goals and Plans

In the coming year, there are several areas in which we expect to expend our efforts on the ONCOCIN System:

- 1) To generalize the reasoning and interaction components of the ONCOCIN system for other applications.
- 2) Extensions and generalizations of the OPAL and PROTÉGÉ knowledge acquisition experiments
- 3) Extensions to the strategic planning framework including research on temporal representations, mathematical modeling and summarization
- 4) To determine and begin implementation of ONCOCIN on more accessible hardware platforms..

B. Justification and Requirements for Continued SUMEX Use

All of our research takes place in the context of the SUMEX resource. The development of our project will continue to take place on LISP machines which we have purchased or which have been donated by the XEROX Corporation, with a gradual transition to Mac II's. The word processing and communication requirements will be met by the Mac II's plus the excellent network services provided by SUMEX.

C. Requirements for Additional Computing Resources

Most of our CPU needs are met with our current equipment. However, our requirements for high bandwidth communication facilities are increased as we have an increasingly distributed environment. We would appreciate support that provides equivalent resources to what we are used to on the DEC-20 including mail support, and file servers. We will continue to need access to the international networks that we now use for much of our communication with colleagues. For example, Mark Musen (who recently finished his Ph.D., and is now an Assistant Professor in our department) was in Holland for 9 months. During this period, we were in constant electronic communication with him over a set of networks. This has been a significant asset to our research project.

D. Recommendations for Future Community and Resource Development

The continuation of network support and file service are our major needs. Help with software selection and implementation for our Mac II's is also an important requirement to maintain research continuity in our heterogeneous environment. Maintaining the excellent mail service and access to the international networks is also essential for our research.

IV.A.4. PENGUIN Project

The PENGUIN Project: Toward Expert Database Systems in Biomedicine

Gio Wiederhold, Ph.D.
Departments of Computer Science and Medicine
Stanford University

Thierry Barsalou, M.D.
Medical Computer Science
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

Crucial problems encountered in the management and structuring of large quantities of knowledge are being addressed by the KBMS/KSYS project, primarily under DARPA sponsorship. The RADIX project (phased out last year) on automated knowledge acquisition from large medical databases has provided valuable experience for the KBMS/KSYS group. The KBMS/KSYS work, in turn, is leading to another important medical application: PENGUIN, which investigates combining knowledge-base and database technology into so-called expert database systems. This project was initially supported by KBMS, but is now fully funded by a grant from NLM.

A. PENGUIN Project — Rationale

Databases and expert systems share a common goal — generating useful information for action — but accomplish their tasks separately, using different principles. It is clear, however, that future information systems will require both the problem-solving capabilities of expert systems (ES's) and the data-handling capabilities of database management systems (DBMS's). Indeed, combining database and expert system technologies into expert database systems (EDS's) is an emerging research area. One can define an EDS as "a system for developing applications requiring knowledge-directed processing of shared information". From a perspective of developing advanced biomedical information systems, this definition conveys two precise scenarios: (1) enhancing DBMS's with structuring and manipulation tools that take more semantics into account; (2) allowing ES's to access and to handle efficiently information stored in database(s).

The object-oriented paradigm has gained much attention in recent years. In the database field, object-oriented DBMS's have emerged. The concept of an entity is also widely used by database design tools. In the field of artificial intelligence, frames are a well-known knowledge representation scheme. Although frames were conceived separately from the object paradigm, the two are in fact consistent with each other. In this research project, called PENGUIN, we investigate the hypothesis that the object-oriented approach can also serve as a unifying scheme for developing EDS's.

We have the opportunity to explore this hypothesis in a practical biomedical environment. Fluorescence Activated Cell Sorting (FACS) is emerging as a major source of information for biomedical research and clinical practice. Currently, achieving good FACS performance requires analyzing and integrating various complex data and knowledge sources. PENGUIN is thus aimed at developing methods for bringing together expert system and database technologies in an integrated advice system that could fulfill the information needs of FACS investigators. Central to this work is a very close collaboration between researchers in the Medical Information Sciences Program and the Departments of Computer Science and Genetics.

B. PENGUIN — Medical Relevance and Collaboration

FACS has already demonstrated significant promise in clinical as well as research-oriented areas. As indicated earlier, basic studies identifying lymphocyte subpopulations have now been translated into clinical applications including monitoring of AIDS patients. Similar FACS applications can now be found in such diverse specialties as Hematology, Oncology, Infectious Diseases and even Gynecology-Obstetrics where current studies are evaluating FACS analysis of maternal blood sample as a potential non-invasive method for prenatal diagnoses.

The major block to proliferation of these kinds of valuable studies involves the requirement for greater skills in reagent selection and FACS machine operation than are typically available in basic and clinical research settings.

Thus developing sophisticated computer tools that provide a new level of automatic analysis and control for FACS and greatly facilitates FACS use by reducing the need for on-site human expertise should significantly improve the potential for using this versatile methodology in biomedical research and clinical practice.

C. PENGUIN — Highlights of Research Progress

Our current effort focuses on developing a computer-based advisory system to assist the FACS investigator in designing experiment protocols. As mentioned above, this system will combine database and artificial intelligence methodologies in an integrated framework, as to provide 1) several levels of abstraction for information management and retrieval from a relational database system, 2) access to and integration of the results of past similar experiments as additional units of information through an interface with existing databases of FACS data and 3) inference capabilities coupled to the database for high-level interactions with scientists during the design process. Although this work is motivated by a specific application, the formal design of the system will be domain-independent; it is then our belief that ideas, principles and programs developed in this process will be applicable to other medical and non-medical areas.

In the past two years, we have developed the concept of an object-based interface on top of a relational database system and implemented an initial

prototype of the interface. We have drawn an analogy between the notions of object and database view. Using this analogy, we have defined three components in the object interface:

- 1) The object *generator* maps relations into object templates where each template can be a complex combination of join (combining two relations through shared attributes) and projection (restricting the set of attributes of a relation) operations on the base relations. In addition, an object network groups together related templates, thereby identifying different object views of the same database. The whole process is knowledge-driven, using the semantics of the database structure. We define the object schema as the set of object networks constructed over a given database. Like the data schema for a relational database, the object schema represents the domain-specific information needed to gain access to PENGUIN's objects; this information enables us to combine well-organized, regular tabular structures — the relations — into complex, heterogeneous entities — the objects.
- 2) The object *instantiator* provides nonprocedural access to the actual object instances. First, a declarative query (e.g., Select instances of template x where attribute y < 0.5) specifies the template of interest. Combining the database-access function (stored in the template), and the specific selection criteria, PENGUIN automatically generates the relational query and transmits it to the DBMS, which in turn transmits back the set of matching relational tuples. In addition to performing the database-access function, the object template specifies the structure and linkage of the data elements within the object. This information is necessary for the tuples to be correctly assembled into the desired instances. Those instances are then made available to the expert system directly, or to the user through a graphic interface.
- 3) The object *decomposer* implements the inverse function; that is, it maps the object instances back to the base relations. This component is invoked when changes to some object instances (e.g., deletion of an instance, update of some attributes) need to be made persistent at the database level. An object instance is generated by collapsing (potentially) many tuples from several relations. By the same token, one update operation on an object may result in a number of update operations that need to be performed on the base relations.

In addition to augmenting the relational model with an object layer, we have devoted a significant portion of our effort to user-interface issues, as one of the main challenges faced by designers of information systems is the development of high-grade interfaces. Indeed, wide acceptance and use of any computer program presuppose a good interface. Our work follows from two basic principles. First, domain-specific interfaces should be separated from all other aspects of the system, which could then offer a number of interfaces tailored to the problem at hand. Second, some intelligence should be built into the interfaces to make them truly cooperative, active agents

instead of simply passive media. Our research hypothesis is that a hypertext approach can facilitate the design of such intuitive user interfaces for decision-support systems. We are exploiting the HyperCard authoring tool for that purpose.

Using HyperCard, one can design, prototype, and debug a tailored interface in a single session—interface that takes full advantage of a visual representation of the underlying database (through, for instance, icons, buttons, and graphics). Within the taxonomy of direct-manipulation interfaces, PENGUIN employs visual correlation to specify objects and actions. Inasmuch as visual-correlation methods require the user to recognize rather than to recall the appropriate command name, they typically place a lesser burden on human memory than command languages do. Most important, perhaps, refinement of the interface can occur without writing a line of code through the large palette of tools that we have assembled.

Preliminary results of PENGUIN's implementation indicate that an object-based architecture provides (1) efficient access to and manipulation of complex units of biomedical information while preserving the advantages associated with persistent storage of data in relational format, and (2) a domain-independent, bidirectional communication channel between relational database systems and expert systems.

The latter result is based on our ongoing work toward integrating PENGUIN and KNET. Developed in our laboratory by R. Martin Chavez, KNET is a general-purpose environment for constructing probabilistic, knowledge-intensive systems based on belief networks and decision networks. Integration of PENGUIN and KNET requires translating KNET knowledge bases into relational format. For that purpose, we have defined a general relational schema that can accommodate any domain-specific KNET knowledge base. Thanks to that general schema, the translation process is for the most part automated. We now need to establish communication between the two systems. Because KNET is object-oriented, PENGUIN's object layer represents a natural communication channel. We therefore envision a tripartite architecture with a database layer and a knowledge-based layer connected by the object layer.

Finally and in parallel to system development, we are currently engaged in a knowledge acquisition process to define the structure of the database and of the knowledge base. Through interviews with our experts in the Genetics department, we are eliciting the structure of the various components (genetic, histological, and serological information) of the FACS reagents database.

D. Publications of the PENGUIN project

- 1) Barsalou, T. and Wiederhold G.: "A cooperative hypertext interface to relational databases"; submitted to the Thirteenth Annual Symposium on Computer Application in Medical Care, March 1989.

- 2) Barsalou, T., Chavez, R.M. and Wiederhold G.: "Hypertext interfaces for decision-support systems: A case study"; To appear in the Proceedings of MEDINFO '89, IFIP, Beijing, China, October 1989.
- 3) Barsalou, T. and Wiederhold G.: "Knowledge-based mapping of relations into objects"; To appear in Computer Aided Design, 1989.
- 4) Barsalou, T.: "An object-based architecture for biomedical expert database systems"; in R.A. Greenes (editor), Proceedings of the Twelfth Symposium on Computer Applications in Medical Care, pages 572—578, IEEE Computer Society Press, Washington, D.C., November 1988.
- 5) Barsalou, T. and Wiederhold G.: "Applying a Semantic Model to an Immunology Database"; in W.W. Stead (editor), Proceedings of the Eleventh Symposium on Computer Applications in Medical Care, pages 871-877, IEEE Computer Society Press, Washington, D.C., November 1987.
- 6) Barsalou, T., Moore, W.A., Herzenberg, L.A. and Wiederhold G.: "A Database System to Facilitate the Design of FACS Experiment Protocols" (abstract); Cytometry, Vol.97, August 1987.
- 7) Law, K.H. and Barsalou T.: "Applying a semantic structural model for engineering design"; To appear in the Proceedings of the Computers in Engineering Conference, American Society of Mechanical Engineers, Anaheim, CA, August 1989.
- 8) Wiederhold, G., Rathmann, P.K., Barsalou, T., Lee, B.S. and Quass D.: "Partitioning and composing knowledge"; Submitted to Information Systems, 1989.
- 9) Wiederhold, G., Barsalou, T. and Chaudhuri S.: "Managing objects in a relational framework"; Technical report No. STAN-CS-89-1245, Computer Science Department, Stanford University, January 1989.
- 10) Wiederhold, G.: "Hospital Information Systems"; in Encyclopedia of Medical Devices and Instrumentation, vol.3, Webster, John G.(ed), Wiley 1988, pp.1517—1542.
- 11) Wiederhold, G.: "Sharing and partitioning large knowledge bases"; in Appelrath, Cremers, and Schiltknecht (eds), 'PROTOS, Prolog tools for building expert systems', EUREKA report EU56, Basel, Switzerland, pp. 77—83, December 1988.
- 12) Wiederhold, G.: *File Organization for Database Design* McGraw-Hill Book Company, 1988.

- 13) Wiederhold, G., Michael G. Walker, Waqar Hasan, Surajit Chaudhuri, Arun Swami, Sang K. Cha, XiaoLei Qian, Marianne Winslett, Linda DeMichiel, and Peter K. Rathmann: "KSYS: An Architecture for Integrating Databases and Knowledge Bases"; Computer Science Department, Stanford University, May 1987; in Amar Gupta and Stuart Madnick (editors) 'Technical Opinions Regarding Knowledge- Based Integrated Information Systems Engineering', MIT, 1987,
- 14) Wiederhold, G.: "Knowledge versus Data"; Chapter 8 of 'On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies' (Brodie, Mylopoulos, and Schmidt, eds.), Springer Verlag, June 1986, pages 77 to 82.
- 15) Wiederhold, G., Blum, R.L., and Walker M.: "An Integration of Knowledge and Data Representation"; Proc. of Islamorada Workshop, Feb.1985, Computer Corporation of America, Cambridge MA; Chapter 29 of 'On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies' (Brodie, Mylopoulos, and Schmidt, eds.), Springer Verlag, June 1986, pages 431 to 444.
- 16) Wiederhold, G.: *Views, Objects, and Databases*. IEEE Computer 19(12):37-44, December, 1986.
- 17) Missikoff, M., and Wiederhold G.: *Towards a Unified Approach for Expert and Database Systems*. In 'Expert Database Systems', Larry Kerschberg (editor), Benjamin/Cummings, 1986, pages 383-399; also in Proceedings of First Workshop on Expert Database Systems, Kiawah Island, South Carolina, Oct. 1984, vol.1, pp.186-206.
- 18) Wiederhold, G.: *Knowledge Bases*; Future Generations Computer Systems, North-Holland, vol.1 no.4; April 1985, pp.223—235.
- 19) Wiederhold, G.: *Database Design* (in the Computer Science Series); McGraw-Hill Book Company, New York, NY, May 1977, 678 pp. Second edition, Jan. 1983, 768 pp.
- 20) Wiederhold, G.: In D.A.B. Lindberg and P.L. Reichertz (Eds.), *Databases for Health Care*, Lecture Notes in Medical Informatics, Springer-Verlag, 1981.
- 21) Wiederhold, G.: "Database technology in health care"; J. Medical Systems 5(3):175-196, 1981.

E. Funding Support Status

1987-1990 Principal Investigator: Gio Wiederhold 25% effort
 FACS-Penguin: An Expert Workstation for Flow Cytometry
 (NLM/NIH : \$856,449)

1987-1991 Principal Investigator: Gio Wiederhold 10%
 Support for Parallel Design in an Engineering Information System
 (NSF DTMP 8619595 \$153,766, \$146,360 prop.,\$156,044 prop.)

1987-1988 Principal Investigator: Gio Wiederhold 5%
Validation of Knowledge from a Database
(IBM KBS Menlo Park, \$50,051)

1986-1990 Principal Investigator: Gio Wiederhold 50% effort
Knowledge Base Management Structures
(ONR/SPAWAR/ARPA,N39-84-C-211, task 7: \$1,756,410)

1987-1988 Principal Investigator: Gio Wiederhold 5%
Reasoning about R1ME
(Digital Equipment Corp: \$131,337)

1987-1990 Associate Investigator: Gio Wiederhold 10% effort
Integrating Knowledge and DBMS
(SRI AI/ONR/SPAWAR/ARPA: \$35,003 Stanford Subcontract)

1988-1990 Principal Investigator: Gio Wiederhold 10% effort
PARADATA: Databases on Parallel Computers
(NSF/IRI/K&DSP, year 1: \$148,116, expected year 2 \$159,689)

1988 Principal Investigator 10% effort
DADAISM: DBMS in and for ADA-supported Information System
Management
(SRI/NRL/STARS/DoD: supplement \$30,000)

1988 Principal Investigator 20% summer, 10% academic year
(6 months) DADAISM: DBMS in and for ADA-supported Information
System Management
(SRI/NRL/STARS/DoD: \$190,861)

1988-1991 Principal Investigator: 5% effort
Computer-Assisted Analysis of Auroral Images Obtained from
High Altitude Polar Satellites
(CESDIS/USRA/NASA, approved,\$577,254 requested, being funded at
approx. \$500,000, with supplementary support for Dr. Robert Clauer)

1988 Co-investigator with Kincho Law 5% effort
Object Databases for CAD/CAE data
(Stanford/CIFE, \$34,500)

1990-1994 Associate Investigator, Paul Losleben (PI) 10% + 10% effort
Micro Factory Project, Database for Manufacturing Expert,
Systems Process Design Database,
(Texas Instruments/NSF, \$287,442 + \$288,877 out of \$4,540,547)

Requests in process

1989-1994 Consultant Investigator, James Fries (PI) 5% effort
ATHOS: AIDS- Time-oriented Health Outcome Study;
(NCHSR/PHS/HEW, \$7,562,601 requested)

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A. Collaborations

SUMEX AIM provides the central communication node for our research. While individual experiments tend to take place on workstations, the availability of SUMEX to load and edit files, communicate with colleagues and peers is absolutely critical.

B. Interactions with Other SUMEX-AIM Projects

During the current reporting year we have had frequent interaction with members of other SUMEX projects; for example, presentation of research results at Stanford Medical Information Science Colloquia, discussions of automated discovery and automated summarization, practical programming issues, and training of Medical Computer Science Students in the use of KEE, Lisp workstations, and so on. The SUMEX community is an invaluable resource for providing such interaction.

C. Critique of Resource Management

The new Sun-4 system provides acceptable performance. The archiving facility (SAFE) seems to be very loaded, so that we are forced to make decisions on retention of past material more frequently than optimal. As users of large databases this is one of the resources upon we must draw frequently.

The SUMEX resource management continues to be accessible and quite helpful. New networking links, for instance to the Genetics Department equipment, make new collaborations viable.

III. RESEARCH PLANS

A. Project Goals and Plans

The long-range goal of the PENGUIN project is to integrate our experiment design advisory system with the set of programs developed in the Genetics department for FACS operation and control, thereby defining a comprehensive "FACS Workstation" that should considerably facilitate effective utilization of FACS technology. In the shorter term, we will:

- 1) complete the development and testing of the object interface. The interface should then be made a fully domain-independent module that could be applied in different contexts. One such context is civil engineering where the concepts developed in PENGUIN are being applied to engineering design problems.
- 2) integrate the PENGUIN and KNET systems, so that PENGUIN can act as an intelligent data server for KNET.

- 3) validate the structure of the database and make it available to the scientists in the Genetics department, for further refinement as well as for routine data storage and retrieval.
- 4) move fully toward a distributed environment with Macintoshes as local workstations connected to a central VAX computer, acting as a database server.
- 5) develop the expert system module which will be coupled to the database through the object interface. The expert system module is aimed at assisting people in an experimental design task. This can actually be done by taking two different approaches, one being the actual automatic design of the experiment (i.e., a planning task), the other being the critique of a protocol proposed by the user. Although we recognize both directions as important, we will focus on the first approach because knowing how to properly plan new experiments is a necessary step prior to any critiquing process. Although the problem solving strategies of our experts are not yet fully determined, we plan to investigate the general concept of hierarchical planning.

B. Justification and Requirements for Use of SUMEX

The PENGUIN project is dependent on new methods of communication among clinical and academic researchers.

C. Recommendations for Resource Development

The movement toward workstations is obviously the direction of the future. We hope that the communication capability, now supported via the central SUMEX facility will continue to be supported. We continue to make constant use of this communication with our colleagues within and outside of Stanford, preparation of large documents, file and database handling, and program demonstrations.

IV.A.5. PROTEAN Project

Oleg Jardetzky
Nuclear Magnetic Resonance Lab, School of Medicine
Stanford University

Bruce Buchanan, Ph.D.
Computer Science Department
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

The goals of this project have been related both to biochemistry and artificial intelligence: (a) use existing AI methods to aid in the determination of the 3-dimensional structure of proteins in solution, and (b) use protein structure determination as a test problem for experiments with large scale constraint satisfaction, an area of increasing interest to artificial intelligence. Empirical data from nuclear magnetic resonance (NMR) and other sources may provide enough constraints on structural descriptions to allow protein chemists to bypass the laborious methods of crystallizing a protein and using X-ray crystallography to determine its structure. This problem exhibits considerable computational complexity, but by formulating it as search problem, it should be possible to utilize many of the knowledge-based techniques developed in AI for dealing with large search spaces.

B. Medical Relevance

The molecular structure of proteins is essential for understanding many problems of medicine at the molecular level, such as the mechanisms of drug action. Using NMR data from proteins in solution will allow the study of proteins whose structure cannot be determined with other techniques, and will decrease the time needed for the determination.

C. Highlights of Progress

With the departure of Prof. Bruce Buchanan to the University of Pittsburgh, work on the PROTEAN project is nearing conclusion and has focussed on the completion of thesis work by Bruce Duncan and Russ Altman.

The core strategy of this project for assembling protein structures involves functionality at the atomic as well as the more abstract solid level, and demonstrates an application of our general approach, which we call *heuristic refinement*.

The heuristic refinement method falls within an *exclusion paradigm* for interpreting protein structure. In this paradigm all the atoms in the protein are initially assumed to be everywhere with respect to some arbitrary coordinate system. As constraints are introduced these initially infinite *accessible volumes* are gradually reduced until the remaining accessible

volumes are small enough so that a representative set of structures may be enumerated.

The exclusion paradigm has not been tried previously because it appears to be computationally intractable for any reasonably sized protein. For this reason most current techniques (non AI-based) use an *adjustment paradigm* to optimize a single structure towards a global minimum. These techniques all share difficulties common to optimization, namely lack of convergence for large numbers of variables or for starting structures far from the global minimum. In addition, for reasons of computation time, they are not able to provide a representative sample of all structures compatible with the constraints.

The exclusion paradigm, by systematically eliminating structures incompatible with the constraints, has the advantage that all, or at least a representative set, of structures compatible with the constraints are retained. In order to deal with the combinatorics the heuristic refinement method formulates the problem as one of search, which is the fundamental paradigm of artificial intelligence. In particular, the problem is formulated as a geometric constraint satisfaction problem (GCSP), which can be solved by backtrack search if the size of the atomic accessible volumes can be made small enough.

The complexity of solving a GCSP depends on the number of objects and the number of possible locations that each object can have in space. Therefore, the heuristic refinement method utilizes four techniques to reduce these two numbers to the point that backtrack search may be applied. The effect of each of these techniques is to utilize knowledge of protein structure and of techniques for solving GCSP's to prune the search tree:

- 1) **Problem decomposition.** The protein is broken into logical subparts such as secondary structures and sidechains. Each of these is treated as a separate GCSP, which is partially solved. Solutions to the subproblems either determine constraints at more abstract levels or are combined into larger subproblems.
- 2) **Problem abstraction.** Groups of locally highly constrained atoms are represented as single abstract "solid" level objects. Solutions to atomic level GCSP's are used to create abstract "solid" level constraints, resulting in a solid level GCSP with far fewer objects. Solutions at the solid level are then expanded to the atomic level, resulting in fewer possible locations for the atoms than would have been present if the initial solid level processing had not occurred. Almost all the work reported in previous reports was at this abstract solid level, but it is the current refinement to the atomic level that is of most interest to biochemists.
- 3) **Local satisfaction of constraints.** In each local GCSP filtering operations called network consistency algorithms are applied, allowing the accessible volumes to be reduced by looking at constraints pairwise (or to higher

order) rather than all at once. These operations, which were initially developed for computer vision, are becoming more popular in AI because of their utility in many forms of constraint satisfaction problems.

- 4) Heuristic control. At each point in the problem solving there are many possible constraint satisfaction operations that may be applied. If there is no bias then the order of operations should not matter, but the efficiency may vary greatly. Much of the previously reported work in PROTEAN went towards developing the BB1 framework for using heuristics to opportunistically determine the best operation to apply at any given time. The BB1 framework has been used to control the solid abstract level of problem solving, but it has not yet been integrated with the atomic level functionality, which is currently manually controlled.

Atomic level constraint satisfaction operations. As part of his Ph.D. work Bruce Duncan has extended many of the solid level constraint satisfaction operations to the atomic level. The computations make use of three different computational techniques: the systematic search of conformational space, a probabilistic updating technique, and a random search (Monte Carlo) technique.

The systematic search procedures use constraint satisfaction techniques and attempt to completely sample the protein conformational space. This is done by examining the positions of small groups of atoms to delete positions that do not satisfy the applied constraints. This is an extension of the techniques used by the solid level of the PROTEAN system.

Since there the primary limitation of the systematic search technique is that not enough atom positions can be simultaneously examined, the probabilistic technique attempts to infer the likely atom positions. In this technique, the atom positions are represented as sampled spatial probability distributions. Distance constraints are also modeled as probability distributions. An updating procedure modifies the spatial distributions of the atoms using the the constraint distribution. This procedure is related to the probabilistic methods developed by Russ Altman. Heuristic constraints on the structure were determined by analyzing known structures.

The Monte Carlo technique computes instances of the protein conformation that span the allowed range. Although probabilistic techniques can be used to compute structures represented by the mean atomic positions and the atomic covariance matrices, detailed analysis requires that individual instances are produced. At the atomic level, the technique of systematically searching the protein conformational space is ineffective to compute instances of the protein conformation. To overcome this limitation, techniques based on random sampling of the conformational space are being developed. This procedure makes use of several computational techniques used by the atomic level system such as simplified representations and the application of constraint satisfaction techniques. These techniques are effective because at this stage of the computation, the protein conformational space has been approximately determined. This technique is has the advantage that the

atomic positions are selected based on a random number generator thus ensuring that an unbiased set of conformations are computed.

This work has made use of the three SUMEX SUN 3/75 workstations (clienta, clientb, clientc) and the SUN file server (KNIFE). An interesting feature of the system is that it can make use of the three SUN workstations operating in parallel (although the interface is a bit clumsy). The code is written in the C language or in the C-shell UNIX command interpreter.

Probabilistic operations at the atomic level. Russ Altman, as part of his Ph.D. thesis, has been working to develop an alternative method for representing the accessible volumes of atoms. Currently, we represent accessible volumes as lists of xyz locations sampled on a regular grid at a specified resolution. In Russ's approach accessible volumes are represented as probability density functions described by a mean and covariance matrix. Constraints are also represented by a mean and variance, and a Kalman filter, which is basically another form of Bayes' formula, is used to determine the reduction in accessible volume. This approach has been used to reconstruct the amino acid tyrosine, and is being extended to handle larger numbers of atoms.

D. Relevant Publications

- 1) Altman, R. and Jardetzky, O.: *New strategies for the determination of macromolecular structures in solution.* Journal of Biochemistry (Tokyo), Vol. 100, No. 6, p. 1403-1423, 1986.
- 2) Altman, R. and Buchanan, B.G.: *Partial Compilation of Control Knowledge.* Proceedings of the AAI, pp 399-404, 1987.
- 3) Altman, R., Duncan, B., Brinkley, J., Buchanan, B., Jardetzky, O.: *Determination of the spatial distribution of protein structure using solution data,* Proceedings of the Alfred Benzon Symposium 26: NMR Spectroscopy and Drug Development, Copenhagen, 1987.
- 4) Brinkley, J., Cornelius, C., Altman, R., Hayes-Roth, B. Lichtarge, O., Duncan, B., Buchanan, B.G., Jardetzky, O.: *Application of Constraint Satisfaction Techniques to the Determination of Protein Tertiary Structure.* Report KSL-86-28, Department of Computer Science, 1986.
- 5) Brinkley, James F., Buchanan, Bruce G., Altman, Russ B., Duncan, Bruce S., Cornelius, Craig W.: *A Heuristic Refinement Method for Spatial Constraint Satisfaction Problems.* Report KSL 87-05, Department of Computer Science.
- 6) Brinkley, J.F., Altman, R.B., Duncan, B.S., Buchanan, B.G., and Jardetzky, O.: *The heuristic refinement method for the derivation of protein solution structures: Validation on cytochrome-b562,* KSL Technical Report 88-03, 1988.

- 7) Brugge, J.A., Buchanan, B.G., and Jardetzky, O.: *Toward automating the process of determining polypeptide secondary structure from ^1H NMR data*, To be published in J. Computational Chemistry, 1988.
- 8) Buchanan, B.G., Hayes-Roth, B., Lichtarge, O., Altman, A., Brinkley, J., Hewett, M., Cornelius, C., Duncan, B., Jardetzky, O.: *The Heuristic Refinement Method for Deriving Solution Structures of Proteins*. Report KSL-85-41. October 1985.
- 9) Duncan, B., Buchanan, B., Hayes-Roth, B., Lichtarge, O., Altman, R., Brinkley, J., Hewett, M., Cornelius, C., and Jardetzky, O.: *PROTEAN: A new method for deriving solution structures of proteins*, Bull. Mag. Res., 8:111-119, 1987.
- 10) Garvey, Alan, Cornelius, Craig, and Hayes-Roth, Barbara: *Computational Costs versus Benefits of Control Reasoning*. Report KSL 87-11, Department of Computer Science.
- 11) Hayes-Roth, B.: *The Blackboard Architecture: A General Framework for Problem Solving?* Report HPP-83-30, Department of Computer Science, Stanford University, 1983.
- 12) Hayes-Roth, B.: *BB1: An Environment for Building Blackboard Systems that Control, Explain, and Learn about their own Behavior*. Report HPP-84-16, Department of Computer Science, Stanford University, 1984.
- 13) Hayes-Roth, B.: *A Blackboard Architecture for Control*. Artificial Intelligence 26:251-321, 1985.
- 14) Hayes-Roth, B. and Hewett, M.: *Learning Control Heuristics in BB1*. Report HPP-85-2, Department of Computer Science, 1985.
- 15) Hayes-Roth, B., Buchanan, B.G., Lichtarge, O., Hewett, M., Altman, R., Brinkley, J., Cornelius, C., Duncan, B., and Jardetzky, O.: *PROTEAN: Deriving protein structure from constraints*. Proceedings of the AAAI, 1986, p. 904-909.
- 16) Jardetzky, O.: *A Method for the Definition of the Solution Structure of Proteins from NMR and Other Physical Measurements: The LAC-Repressor Headpiece*. Proceedings of the International Conference on the Frontiers of Biochemistry and Molecular Biology, Alma Alta, June 17-24, 1984, October, 1984.
- 17) Lichtarge, Olivier: *Structure determination of proteins in solution by NMR*. Ph.D. Thesis, Stanford University, November, 1986.
- 18) Lichtarge, Olivier, Cornelius, Craig W., Buchanan, Bruce G., Jardetzky, Oleg: *Validation of the First Step of the Heuristic Refinement Method for the Derivation of Solution Structures of Proteins from NMR Data.*, Proteins: Structure, Function and Genetics, 2:340-358, 1987.

E. Funding Support

Title: Interpretation of NMR Data from Proteins Using AI Methods

PI's: Oleg Jardetzky and Bruce G. Buchanan

Agency: National Science Foundation

Grant identification number: DMB-8402348

Total Award Period and Amount: 2/1/87 - 9/30/89 \$120,000 (includes direct and indirect costs)

Current award period and amount: 2/1/87 - 9/30/89 \$120,000 (includes direct and indirect costs)

III. RESEARCH PLANS

A. Goals & Plans

The AI research phase of the PROTEAN project is being terminated with the completion of thesis work by Bruce Duncan and Russ Altman. In the present stage of the PROTEAN project, most of the effort is being directed towards a successful application of the system in the Stanford Magnetic Resonance Laboratory of Prof. Oleg Jardetzky (SMRL). All of the PROTEAN system calculations will be performed on national supercomputing resources or on mini-supercomputers assembled in the SMRL for this purpose. Additional program development will be done on workstations, communicating via the local area network maintained by SUMEX.

Input data generation and output data processing also require a certain amount of additional computing, briefly described in the following.

1. SOLUTION STRUCTURE DETERMINATION

Input Generation

For simplicity of computation, hydrogen atoms which have no independent degrees of motional freedom are included in pseudo-atoms. "Pseudo-structures" of the twenty common amino acid residues are described by K. Wutrich, M. Billeter and W. Braun in J. MOL. BIOL. (1983) 169,949-961. The inclusion of the specific amino acid pseudo atoms list for a protein is automated.

Initial Mean Positions / Variance-Covariance Matrices

- (i) Initial means and variance-covariance values of the CA atoms are known from a previous Kalman filter update run.
- (ii) The mean positions of the added backbone, and amino acid residue pseudo atoms, are derived randomly from the CA values of (i). The initial variances of the mean positions are consistent with the uncertainty of the atom being anywhere within the protein domain. Zero covariances are assumed initially between any of two atom positions. Initial CA variances are taken from (i).

Generation of Distance Constraints

- (i) Distance Ranges. The set of distance constraints between pseudo atoms to be consistently updated by the Kalman filter mechanism consists of:

Constraints determined by the primary and secondary structure:

1. Covalent bonding within the backbone, within amino acid residues, and of those to the backbone.
2. Nonbonded distances in the backbone determined by covalent angles, e.g. for residue i:

$$\text{CA}(i) \quad \text{N}(i+1) = 2.38 \text{ \AA}$$

$$\text{CA}(i) \quad \text{O}(i) = 2.40 \text{ \AA}$$

$$\text{O}(i) \quad \text{N}(i+1) = 2.27 \text{ \AA}$$

3. Hydrogen bonding within helices.
4. Short range distances of protons in amino acid groups (pseudo-atoms) to the backbone protons which are three or less dihedral angles away.

Long range NMR NOE distance constraints.

- (ii) Variances. The distance constraints variances are evaluated as a sum of the individual pseudo-atoms variances, and the distance variance.

1. A pseudo-atom variance is assumed to follow the variance of a uniform distribution, i.e. (its van der Waals radius (e.g. 1.8 Å for a CH₃ group)/5)**2.
2. Similarly, if a distance range variance is assumed uniform, it is evaluated by ((a-b)**2)/12 where [a,b] is the distance range.

Thus, the variance of a long range NOE distance constraint is assumed to equal 1.1 Å (= 2* ((1.8)**2 / 5) + ((6 - 3)**2) / 12); the distance includes correction for protons, as discussed in the following).

- (iii) Distance Correction. Since NMR NOE distance constraints are observed between protons, short- and long-range NOE distance constraints have to be corrected for the C-H bond length of CA. 1Å.

In order to generate appropriate means and covariances, and the corresponding set of distance constraints for the Kalman filter program, some computational resources are to be utilized.

Output Data Processing

Results of a Kalman filter run have to be evaluated in terms of the structure's consistency with the distance constraints, and with structures obtained by other methods. This evaluation entails some computations, e.g. distance calculations given a set of (x,y,z) coordinates.

BLOCH EQUATION CALCULATIONS

Input Generation

For the input to the bloch equation program, a standard pdb file is utilized for input data generation :

- (a) A data set including only protons is generated.
- (b) Methyl groups are identified and average positions are evaluated.
- (c) A distance matrix is calculated, taking into account only those proton pairs which are less than 6 Å apart.
- (d) The number of all possible NOE distances for each proton is calculated.

Output Evaluation

A large amount of data processing on an output file has to be performed, as is shown in the following example.

Given the NOE output file for the protonated trp repressor, which was recently calculated, the following computations were performed:

- (a) NOE intensities of proton pairs for various mixing times are separated.
- (b) For each mixing time the following NOE's were to be identified:
 - (1) Intensities of HN(i)-HN(i+1) for each i, where i is the residue number.
 - (2) Intensities of each HN(i) other than to other HN's or HO's.
 - (3) Intensities of (2), but also other than those to side chain i.
 - (4) Intensities of (3), but also other than sequential.

B. Justification for Continued SUMEX Use

The communication and other capabilities provided by SUMEX and the SUMEX staff is essential for continued success of such a large and evolving distributed project.

C. Other computing resources

At this time our computational resources are almost adequate as long as we can continue to use the SUMEX-supported local area network. Since PROTEAN is evolving into a distributed problem-solving system, we actively support and require extensive network access capabilities. We believe that the requirements of PROTEAN make it an ideal test problem for many of the distributed systems issues being considered by the SUMEX core research staff.

IV.A.6. Reasoning Under Uncertainty

Pragmatic Approaches to Reasoning under Uncertainty in Medical Decision Support Systems

Principle investigator: E.H. Shortliffe

Project manager: G.F. Cooper

Students: R.M. Chavez, D.E. Heckerman, E.H. Herskovits,

E.J. Horvitz, M. Shwe, and H.J. Suermondt

I. Summary of research program

I.A. Overview

Medicine is replete with uncertainty. In particular, there is uncertainty due to incomplete and inexact scientific models of human health and disease, and there is uncertainty secondary to incomplete and erroneous data about individual patients. Therefore, in general, it is important that computer systems that assist in medical decision making be capable of representing and reasoning with uncertainty. We are exploring the use of probability theory as a representation of uncertainty in medical diagnostic systems. The advantages of using a probabilistic representation, include that probability 1) is mathematically well-defined and has been studied extensively, 2) provides a common, well-established language for communicating uncertainty, 3) allows the combination of subjective probabilities from medical experts with statistics gathered from databases, and 4) is a necessary component of normative decision systems that are based on decision theory. Nonetheless, there are potential problems associated with using a probabilistic representation. Key challenges include developing knowledge acquisition methods and probabilistic inference techniques that are tractable, particularly in light of the large amount of knowledge that must be represented in broad areas of medicine.

We are currently pursuing approaches that we believe hold promise in making practical the probabilistic reasoning in medical expert system. The specific probability representation we are exploring is called a Bayesian belief network. Such networks provide a mechanism for intuitively and efficiently specifying the probabilistic dependencies among diseases, intermediate pathophysiological states, and evidence (e.g., symptoms). In Section I.B we discuss three areas of current belief network research. Section I.B.1 summarizes recent work on methods that enable the computer to control its own probabilistic reasoning processes so that it focuses its potentially scarce computing resources (particularly time) on the most critical diagnostic inference tasks. Section I.B.2 highlights some specific probabilistic inference algorithms that we are developing and evaluating. Here again the critical issue is speed of probabilistic inference. Section I.B.3 discusses techniques we have developed for efficient acquisition of probabilistic knowledge from medical experts.

I.B. Research progress

I.B.1. Reasoning about inference tradeoffs

Recent work on inference tradeoffs has focused on the analysis of techniques for balancing the costs and benefits of decision-theoretic inference. In this work, we use decision-theoretic control of decision-theoretic inference as a model of rational computational decision making. In particular, the computer system continually calculates the expected value of additional computation in deciding whether to make a diagnostic suggestion based on current computation or to continue computing [16]. For example, the amount of time taken by the computer to refine its diagnosis should generally be small when the patient has an obviously life-threatening problem. Nonetheless, the computer must compute long enough to attain a reasonably accurate diagnostic suggestion, otherwise, based on the system's suggestion, the patient might be treated for the wrong underlying condition. Developing computer systems that can dynamically make tradeoffs between computation time and diagnostic accuracy is a difficult problem. We have made use of partial characterizations of decision-theoretic inference to reason about the value of continuing to compute versus making a diagnostic suggestion. We particularly have been studying the value of applying a probabilistic bounding algorithm to diagnostic problems. This work makes use of summary information about the convergence of interval bounds on the probabilities of potential diseases. Related work has focused on using decision-theory to control the construction of a relatively small, context-sensitive decision model from a much larger belief network, so that the small model can be efficiently solved. The QMR-DT knowledge base (see Section I.B.2) is being used as the large belief network.

I.B.2. Efficient probability inference algorithms

During the past year, we have continued to explore algorithms for probabilistic inference using Bayesian belief networks. In particular, we have implemented inference algorithms that are exact, approximate, and special-case. Exact algorithms return precise, accurate probability answers, but these algorithms can be slow when applied to large, complex medical domains. Approximation algorithms tradeoff precision and accuracy for speed, but bound the imprecision and the inaccuracy with explicit error terms. Special-case algorithms take advantage of special properties of some classes of belief networks (such as restrictions on topology) in order to solve inference problems efficiently. During the past year we have begun to evaluate these exact, approximate, and special-case algorithms.

Exact Algorithms

Regarding exact probabilistic inference methods, we have implemented the Lauritzen-Spiegelhalter algorithm. In the process, we have gained insight into the graphical manipulations necessary for efficient belief-network

inference. Also, while implementing the Lauritzen-Spiegelhalter algorithm, we developed an efficient method for finding maximal cliques in a triangulated graph, a necessary step in the application of the algorithm.

The Lauritzen-Spiegelhalter algorithm, in addition to the Pearl algorithm, which we implemented earlier, has been incorporated into KNET, a general knowledge engineering shell for constructing decision-theoretic medical expert systems [2]. We have developed a set of over 100 randomly generated bench-mark belief networks for formal evaluation of the Pearl and Lauritzen-Spiegelhalter algorithms, and comparison with other exact and approximation algorithms. We have commenced formal evaluation of the Pearl and Lauritzen-Spiegelhalter inference algorithms; preliminary results have given us important insights into the network topologies in which each of the algorithms performs more efficiently [12]. We are in the process of developing and implementing a method that allows us to use both algorithms simultaneously in performing belief network inference. We anticipate that the resulting synthetic algorithm will perform more efficiently than either algorithm alone.

We have designed a temporal probabilistic representation and inference method [25]. We have developed a prototype knowledge base and system that reasons about the causes of temporally-qualified symptoms in a patient. Reasoning about time can considerably increase the computational time complexity of probabilistic inference and our current techniques are likely to be inadequate. Thus, we currently are attempting to develop more efficient temporal inference methods.

Special-Case Algorithms

During the past year we have explored belief-network precompilation as a special-case method [27]. Precompiling a belief network consists of performing probabilistic inference for cases that are likely to be encountered and then storing the results, indexed by the evidence of the case. When the system is given a patient case, it first quickly checks to see it has been stored. If so, it returns the probability distribution over the potential diseases almost immediately. If not, it solves the case more slowly with a belief-network inference algorithm. Work to date has demonstrated the feasibility of the general precompilation concept and the feasibility of caching incomplete evidence sets (that is, sets of evidence for which some values are unknown), or evidence sets selected by their relative expected utilities instead of expected joint probabilities. A theoretical justification of this method's success has been derived, and will be expanded.

Research work in the QMR-DT (Quick Medical Reference, Decision-Theoretic) project has explored probabilistic inference on a large medical knowledge base. During the past year, we developed a probabilistic version of the QMR decision-support system for diagnosis in internal medicine. Knowledge of over 600 diseases and 4000 manifestations in the QMR knowledge base is incorporated into the probabilistic interpretation of the knowledge base used

by QMR-DT. In light of the size of the QMR-DT knowledge base, our approach to developing a pragmatic diagnostic system is first to make assumptions to reduce the inferential and representational complexity. We are now in the process of evaluating and incrementally modifying these assumptions, based on the performance of the system against QMR and patient cases with known diagnoses.

We have developed an algorithm that can perform probabilistic inference in QMR-DT in $O(n m^- 2^{m^+})$, where n is the number of diseases (n is approximately 600), m^- is the number of negative findings, and m^+ is the number of positive findings [19]. This development is a significant improvement over the straightforward inference algorithm, which is $O(2^n)$. In practice, the number of positive findings is far less than the number of diseases. The current algorithm, as implemented in LightSpeed Pascal on a Macintosh II, can score cases with 9 positive findings in less than one minute.

Approximation Algorithms

As part of our work on the value of bounding, we have developed and explored the modulation of the completeness of probabilistic inference by decomposing a problem into a set of inference subproblems, and by ordering the solution of these problem components by their expected impact on the overall probabilistic inference task. Our method, called bounded cutset conditioning [28], is a graceful analogue to the method of conditioning for probabilistic entailment in belief networks as developed by Pearl. Preliminary analysis indicates that this method can increase inference speed by a factor of 2 or 3 in some belief networks.

We also have developed a randomized approximation scheme for belief network inference [13]. Given a set of evidential variables, the algorithm computes posterior probabilities, with high probability, to within a prespecified error. The method combines Monte Carlo techniques, area-estimation strategies, and convergence analysis for time-reversible Markov chains.

I.B.3 Probability assessment

KNET is a general-purpose environment for constructing probabilistic medical expert systems [2, 17, 24]. Such networks serve as graphical representations for probabilistic models. KNET differs from other tools for expert-system construction in that it combines a direct-manipulation visual interface with a formal probabilistic scheme for the management of uncertain information and inference. The KNET architecture defines a complete separation between the hypermedia user interface on the one hand, and the representation and management of expert opinion on the other. KNET has been fully implemented and debugged, and currently runs on Macintosh II hardware. The system offers a choice of algorithms, some approximate and others exact, for probabilistic inference. We have used KNET to build consultation systems for lymph-node pathology [20], bone-marrow

transplantation therapy [23], clinical epidemiology [1], and alarm management in the intensive-care unit [12]. KNET imposes few restrictions on the interface design. Indeed, we have rapidly prototyped several direct-manipulation interfaces that use graphics, buttons, menus, text, and icons to organize the presentation of static and inferred knowledge.

In the past year, research on *similarity networks* [20] has advanced in three areas. First, the theory of similarity networks was formalized. In particular, necessary and sufficient conditions for consistency among *local* Bayesian belief networks in the similarity network were derived. In addition, it was proved that the ad hoc procedure for construction the *global* belief networks from the collection of local belief networks is sound in the sense that any assertion of conditional independence implied by the global belief network can be derived from the structure of the local belief networks [26]. Second, the similarity network representation was used to build a knowledge base for the *entire* domain of lymph node pathology [20]. We found that the use of similarity networks greatly simplified the assessment of dependencies among findings. Also, the time to assess probabilities was reduced by approximately a factor of ten [26]. Third, the knowledge base was evaluated using a decision-theoretic metric [6]. We found that the diagnostic accuracy of the new knowledge base was significantly better than the accuracy of the knowledge base constructed without the use of similarity networks. In addition, we found that the new knowledge base performed as well as the expert within the noise level of the experiment .

I.C. Publications

Articles published:

1. R. M. Chavez and H. P. Lehmann, REFEREE: A belief network that helps evaluate the credibility of a randomized clinical trial, In: *Proceedings of the 1988 AAAI Workshop on Artificial Intelligence in Medicine*, Stanford, California, 1988.
2. R. M. Chavez and G. F. Cooper, KNET: Integrating hypermedia and Bayesian modeling, In: *Proceedings of the AAAI Workshop on Uncertainty in Artificial Intelligence*, Minneapolis, Minnesota, August 1988.
3. G.F. Cooper, Computer-based medical diagnosis using belief networks and bounded probabilities, In: P. Miller (Ed.), *Readings in Medical Artificial Intelligence*, (Springer-Verlag, New York, 1988).
4. G.F. Cooper, A method for using belief networks as influence diagrams, *Proceedings of the AAAI Workshop on Uncertainty in Artificial Intelligence*, Minneapolis, Minnesota, August 1988.
5. G.F. Cooper, invited commentary on: Lauritzen, S. and Spiegelhalter, D., Local computations with probabilities on graphical structures and their application to expert systems, *Journal of the Royal Statistical Society*, B, 50, 1988.

6. D.E. Heckerman, An empirical comparison of three scoring schemes, *Proceedings of the AAAI Workshop on Uncertainty in Artificial Intelligence*, Minneapolis, Minnesota, August 1988.
7. E.J. Horvitz, Reasoning under varying and uncertain resource limitations, In: *Proceedings of American Association for Artificial Intelligence*, Minneapolis, MN, August, 1988.
8. E.J. Horvitz, J.S. Breese, and M. Henrion, Decision theory in expert systems and artificial intelligence, *Journal of Approximate Reasoning*, 2: 247-302, 1988.
9. Lehmann, H., Knowledge acquisition for probability-based expert systems, The Symposium on Computer Applications in Medicine, November 1988.
10. H.J. Suermondt, and G.F. Cooper, Updating probabilities in multiply connected belief networks, In: *Proceedings of the AAAI Workshop on Uncertainty in Artificial Intelligence*, Minneapolis, Minnesota, August 1988.

Articles to appear:

11. T. Barsalou, R. M. Chavez, and G. Wiederhold, Hypertext interfaces for decision-support systems -- a case study, *MEDINFO-89*, November, 1989.
12. I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper, The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks, *Proceedings of the Artificial Intelligence in Medicine Conference*, June, 1989.
13. R. M. Chavez and G.F. Cooper, A fully polynomial randomized approximation scheme for the Bayesian Inferencing Problem, *Networks*, 1989.
14. G.F. Cooper, Expert systems based on belief networks -- Current research directions, *Applied Statistical Models and Data Analysis*.
15. G.F. Cooper, The computational complexity of probabilistic inference using belief networks, *Artificial Intelligence*.
16. E.J. Horvitz, G.F. Cooper, and D.E. Heckerman, Reflection and action under scarce resources: Theoretical principles and empirical study, *Proceedings of the International Joint Conference on Artificial Intelligence*, July, 1989.

Articles recently submitted for publication:

17. R. M. Chavez, Hypermedia and randomized algorithms for medical expert systems, The Symposium on Computer Applications in Medicine, November 1989.

18. R. M. Chavez, An empirical evaluation of a randomized algorithm for probabilistic inference, The AAAI Workshop on Uncertainty in Artificial Intelligence, July 1989.
19. D.E. Heckerman, A tractable inference algorithm for diagnosing multiple diseases, The AAAI Workshop on Uncertainty in Artificial Intelligence, July, 1989.
20. D.E. Heckerman, E.J. Horvitz, B. Nathwani, Toward effective normative decision systems: Update on the Pathfinder project, The Symposium on Computer Applications in Medicine, November 1989.
21. E.J. Horvitz, D.E. Heckerman, K.C. Ng, B. Nathwani, Heuristic abstraction in the decision-theoretic Pathfinder system, The Symposium on Computer Applications in Medicine, November 1989.
22. H.J. Suermondt and G.F. Cooper, Probabilistic inference in multiply connected belief networks using loop cutsets, *International Journal for Approximate Reasoning*.
23. H.J. Suermondt and M.D. Amylon, Probabilistic prediction of the outcome of bone-marrow transplantation, The Symposium on Computer Applications in Medicine, November 1989.

Additional research reports:

24. R. M. Chavez, Hypermedia and randomized algorithms for probabilistic expert systems, Ph.D. thesis proposal, March, 1989.
25. G.F. Cooper, E.J. Horvitz, and D.E. Heckerman, A Model for Temporal Probabilistic Reasoning, Report KSL-88-***, Knowledge Systems Laboratory, Stanford University, July, 1988.
26. D.E. Heckerman, Probabilistic Similarity Networks, Ph.D. dissertation, Medical Information Sciences, Stanford University (anticipated date of completion: June, 1989).
27. E.H. Herskovits and G.F. Cooper, Algorithms for belief network precomputation, Report KSL-89-35, Knowledge Systems Laboratory, Stanford University, April, 1989.
28. E.J. Horvitz, H.J. Suermondt, G.F. Cooper, Bounded cutset conditioning: An incremental-refinement approach to inference under uncertain resources, Report KSL-88-36, Knowledge Systems Laboratory, Stanford University, December, 1988.

Presentations:

1. Chavez, A fully-polynomial randomized algorithm for probabilistic inference, An invited colloquium talk at the Center for Research on Computing Technology, Division of Applied Sciences, Harvard University, Cambridge, MA, February 1989.

2. Cooper, Addressing the computational complexity of medical diagnosis, An invited colloquium, talk at the Computer Science Department, Washington University, St. Louis, MO, December, 1988.
3. Horvitz, Some fundamental problems and opportunities from the standpoint of rational agency, " AAI Spring Symposium on Artificial Intelligence and Rational Agency, Stanford University, March, 1989.
4. Horvitz, Inference under scarce resources, An invited colloquium, talk at the Computer Science Department, Washington University, St. Louis, MO, November, 1988.

I.D. Funding Support

NSF Grant IRI-8703710

Pragmatic Approaches to Reasoning Under Uncertainty
1987-1990

U.S. Army Research Office Grant 25514-EL

Computational Techniques for Probabilistic Inference
1988-1991

II. Collaboration

During the past year we have continued to maintain contact with a number of researchers who are interested in the research goals described in Section I. These include Prof. Ross Shachter (Engineering-Economic Systems at Stanford), Prof. Harry Lewis (Division of Applied Sciences, Harvard University), Dr. Randolph Miller (Medical Informatics, University of Pittsburgh), Profs. Peter Szolovits and Ramesh Patil (Clinical Decision Making Group, M.I.T.), Prof. Max Henrion (Carnegie-Mellon University), Dr. David Spiegelhalter (Medical Research Council, England), and Dr. Stig Andersen (Aalborg University, Denmark). The SUMEX-AIM computer resource has greatly facilitated our maintaining communication with these researchers during the year.

The majority of our group participated in the AI in Medicine Spring Symposium held last year.

III. Research plans

III.A. Reasoning about inference tradeoffs

So far, our work on the value of probabilistic inference has focused on the value of computation in the setting of probability bounding algorithms. Theoretical work in the coming year will explore the possibility of tractable value-of-information calculations and approximations for several families of probability distributions. Also, in the coming year, we will work to extend the bounded-conditioning approach to inference under uncertain resource constraints to handle multiple pieces of evidence. We will seek to make the algorithm more robust by combining the method with a stochastic simulation

approach for determining weights on alternative instances. We also plan to construct an inference simulation system for gathering information about the convergence of inference algorithms under different evidential settings. The goal of the off-line simulation is to collect parameters that can be used in real-time value of computation calculations for controlling inference. Finally, we plan to evaluate the decision-theoretic control of the construction of relatively small decision models from a large homogeneous belief network, in the internal medicine (QMR-DT knowledge base) setting.

III.B. Efficient probability inference algorithms

We plan to continue the formal evaluations of the algorithms that we have implemented, and submit the results for publication. We intend to apply the results of these evaluations in our work on combining algorithms for inference in a given belief network, and on developing new algorithms.

We will extend our current work on temporal belief-network representations and attempt to design more efficient methods for temporal probabilistic inference.

Other work will include refining and designing algorithms for exact and approximate inference on the QMR-DT belief-network. In particular, research in this area will focus on testing the validity of assumptions in the QMR-DT knowledge base and relaxing the sensitive assumptions by augmenting the inference algorithms or the knowledge base.

III.C. Probability assessment and knowledge acquisition

The QMR-DT knowledge base currently consists of a two-level belief network in which diseases are said to cause findings. We plan to augment the QMR-DT knowledge base to more realistically model the causal structure linking findings, diseases, and intermediate pathophysiological states.

During the summer of 1989 we will complete our work on similarity networks, which are used to increase the efficiency of acquiring probabilities directly from experts. We will turn our efforts toward developing a system that uses a medical database and a set of initial constraints to automatically construct a belief network.

IV.A.7. VentPlan Project

VentPlan Project: Combining Quantitative and Qualitative Techniques for Data Interpretation and Therapy Planning in the Intensive Care Unit

Lawrence Fagan, M.D., Ph.D.
Department of Medicine
Stanford University

Adam Seiver, M.D.
Department of Surgery
Veterans Administration Hospital
Palo Alto, California

Lewis Sheiner, M.D.
Department of Laboratory Medicine
University of California, San Francisco

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

We are designing a data-interpretation and therapy-planning system for the intensive care unit (ICU). Fundamental research issues in temporal reasoning are associated with the ICU application area including assimilation of incoming data, representation of time-oriented intervals, and description of ongoing physiological processes [Fagan 84]. In addition, in ICUs of the 1990's, many more physiological measurements will need to be collected at frequent intervals, and increased access to the current medical record in coded format will be possible. The goal of our system is to gather multiple measurements as they become available in the ICU, determine the meaning of the measurements with reference to the particular clinical context and the patient's individual response patterns, and to suggest alternative settings for the mechanical ventilator that supports the patient's respiratory function following surgery.

Our approach is to coalesce quantitative and qualitative models. Researchers have built elaborate mathematical models of the respiratory and cardiac systems, but it is impractical to base the entire reasoning process on complex equations. These models may take too long to process data or make assumptions about the clinical situation that are unwarranted. There exist many ways to represent the static relationships between different measurements and their related diagnoses, but those systems cannot easily represent complex temporal information. We wish to combine the qualitative techniques in order to build a system that utilizes the advantages of each type of model.

Adapting our approach for planning treatments for cancer patients with unusual clinical courses [Langlotz 87], we will use strategic knowledge to create patient-specific specializations of standard treatment plans. These

plans will be used as general guidelines by the mathematical model to start a search for optimal treatment interventions. In a similar manner, a Bayesian net will be used to specify the initial ranges for parameters used by the mathematical model as it attempts to adapt to the the incoming data. The Bayesian net can also be used to provide a type of "smart alarms" where the interpretation of data is influenced by the specific diagnoses attributed to the patient. We will use decision analytic methods to evaluate and explain the various treatment options available at any point in time. The long-term goal of this project is to embed the decision-making components within the data management tasks of the ICU.

B. Medical Relevance and Collaboration

The problem of too much data being generated in the ICU is well recognized. Originally, monitors were designed to provide more objective assessments of the physiology of the patients in life-threatening situations. However, as more and more measurements became available, the ability of clinicians to assimilate the data began to drop. Expert systems can be designed to sort through the data, recognize untoward events in context and help with therapy selection. An early version of this was the VM system, which was based on extensions to the production rule framework. The current research has far broader goals, including real-time response using multiple methods for reasoning and the use of integrated mathematical models. This has led to a three way collaboration between the VA hospital which is installing a data management system for a new Surgical ICU (Seiver), the Medical Computer Science Group where investigations in qualitative-quantitative reasoning is taking place (Fagan), and the mathematical modeling group at U.C.S.F (Sheiner). In addition, we are continuing informal discussions with Barbara Hayes-Roth's group (see description of Guardian project).

C. Highlights of Research Progress

We have built a prototype version of the system which is being tested on realistic data derived from ICU measurements. The long range therapy planning components have not yet been created, but we have linked the mathematical models with the Bayesian network and the decision-theoretic component. The entire system is held together with a control algorithm that selects the order that modules show be invoked.

The central design principle of the VentPlan project is to develop a method for combining qualitative and quantitative modeling techniques. *Mathematical models* require the relationships between variables to be expressed as equations. Describing how a particular disease state—such as new onset of pulmonary embolus—affects a variety of respiratory and cardiac parameters is difficult. We are using *probabilistic causal networks* (also known as Bayesian networks) to represent qualitative relationships of the form: if A is present and B is high, then C is low with some probability. A complex set of relationships of this type can be built up into a network.

The mathematical model is able to estimate model parameters using patient data in order to generate patient-specific predictions of model variables—for example, partial pressure of CO₂ in arterial blood (paCO₂). The patient-specific mathematical model can be used to predict future measurements, which is difficult to do using the network formalism. Two other components complete the system. The first component is the *control algorithm*, which examines the incoming data and determines which of the modules should process the information. The *utility model* is used to encode physicians' preferences for treatment goals—for example, representing the tradeoff between providing a sufficiently high level of oxygen to the tissues and causing toxicity from having too high a level of inspired oxygen. VentPlan uses the utility model in the therapy planning process to rank plans that are created by the mathematical model.

VentPlan Control Algorithm. The control algorithm is activated each time that new data become available. Each iteration begins with the acquisition of new data, such as a blood-pressure reading. If the data do not directly correspond to quantities in the physiologic model, the belief network is evaluated. The network takes these quantitative inputs and the qualitative information—for example, the diagnosis—and derives prior-probability distributions for the shared model parameters. The mathematical model updates the parameters by combining the probability distributions with quantitative observations—for example, measurements of oxygen concentration in the blood. The mathematical model is then used to search for an optimal plan. The program simulates therapy plans under consideration by running the model, then ranks them using the plan evaluator. The therapy plans are sequentially modified to find the best plan. The calculated optimal plan is the recommended plan, which is compared to the current plan; this comparison is presented to the user.

VentPlan continuously monitors the stream of data measurements, reapplying this control sequence to refine the model and to recalculate the optimal plan as new information becomes available. New data are compared with the expected values derived from the mathematical model or the belief network; in this way, the program accomplishes expectation-driven analysis of the incoming data.

Belief Nets. Medical knowledge is represented in a belief network as a directed, acyclic graph in which nodes represent domain variables and arcs show important dependencies among these variables. Probabilities are attached to nodes and to groups of arcs. Prior probabilities are considered as static facts in this knowledge base. The conditional probabilities for a node are equivalent to rules of the form: "IF (parent node values) THEN (node values), with a stated probability."

Typical inputs are "mild congestive heart failure," "normal respiratory volume," and "decreased temperature." The belief network module provides probability distributions for the parameters that it shares with the mathematical model based on the information computed by the network. The

corresponding outputs are estimates (mean and standard deviation) for each of the parameters, for example, cardiac output 4.4 +/- 2.3 l/min, and oxygen consumption 180 +/- 120 ml/min.

Diagnosis nodes are at the top of the network. These nodes have no predecessors, and we assume they are mutually independent a priori. All diagnosis nodes are associated with a set of mutually exclusive and exhaustive values representing the presence, absence, or severity of a particular disease. *Measurement nodes* represent any available quantitative information. All continuous variables are represented categorically, with sets of discrete intervals dividing the value range. Depending on the necessary level of detail, three to six categories are used for each node. *Parameter nodes* are inferred entities that cannot be measured directly. Each parameter node corresponds to one of the parameters shared with the mathematical model and with the plan evaluator.

The probabilities in a belief network can represent objective as well as subjective knowledge. The network for the VentPlan architecture contains statistical data on prior probabilities, objective conditional probabilities computed from physiologic equations, and subjective assessments. It is necessary to obtain conditional probabilities for the states of a node given all different states of the parent nodes. A probability editor lets the user browse through this multidimensional probability matrix. An equation editor generates conditional probabilities for any deterministic relationship.

The current VentPlan network consists of 81 nodes. Eight nodes represent diagnoses corresponding to typical problems in ventilator care (for example, airway obstruction, infection, and heart failure); 18 nodes represent input measurements. Information for both types of nodes is placed in associated *report nodes*. These report nodes are influenced by error nodes, which implement error functions. These error nodes allow representation and detection of conflicting evidence. For example, some evidence might suggest that the patient has a high oxygen concentration, whereas other evidence might suggest that the oxygen concentration is low. Both pieces of information can be incorporated; their error functions will regulate the effect on the rest of the network. If the information on the low oxygenation is from a highly erratic sensor, it will be discounted, while information from a more reliable blood-gas analysis will have a greater influence.

The inference engine is based on the Lauritzen–Spiegelhalter algorithm for local probability computations on graphical structures. It finds a set of probability distributions consistent with the available evidence and with the conditional probabilities in the network. The algorithm rearranges a network into a tree structure suitable for fast updates. Details on the algorithm can be found in Appendix F. We assume that the discrete probability distributions are approximated by the normal distribution. This assumption enables the network module to calculate initial parameter estimates as means and variances of the equivalent normal distribution.

Mathematical Model. The mathematical model of the VentPlan prototype consists of a quantitative model and a variety of numerical routines. The model is written as a set of linked differential equations that represent the mechanisms of oxygen transport through the body. Variables in the model equations capture the time-varying concentrations of oxygen as that gas flows through the circulatory system. The model has a number of highly nonlinear aspects, so it requires solution by iterative numerical techniques. The model makes a number of assumptions in order to maintain simplicity, as discussed in the section on levels of model detail.

Parameters are used in the mathematical model to represent the relevant underlying mechanisms of the patient. Patient parameters often are not directly observable, or are costly to measure. Once their values are known, the underlying patient state is largely characterized. For example, we use the parameter cardiac output to represent the volume of blood pumped by the heart per unit time. This parameter is difficult to measure directly, but the modeling procedures can infer its value from other observations.

Inputs to the model consist of initial parameter values, various patient measurements, the ventilator settings (treatment plan), and the times at which the observations were obtained. Solution of the model equations (simulation of the model) gives a prediction about the patient state resulting from these inputs. The model simulation is also used in the tasks of parameter updating and plan optimization. Time is included directly in the system equations (the derivatives are taken with respect to time). Patient observations are time-stamped to allow reasoning about different patient contexts and retrograde fitting of model parameters. Predictions may be generated for any desired future time. If predictions are requested for times in the distant future (with respect to the time constants of the model), the equilibrium form of the differential equations is solved.

Prior-probability distributions for the shared VentPlan parameters are provided by the belief network. When direct observations of the system are scarce, these values form a basis for model predictions. As more measurements become available, the prior-probability distributions become less important. This process of updating system parameters is known as *parameter estimation*. Parameter estimation converts the general mathematical model to a patient-specific model, the latter being able to generate patient-specific predictions. The optimization procedure uses the patient-specific model to find the best plan, given a specific patient state. It accomplishes this task by minimizing an objective (search) function defined by the plan evaluator, which proceeds by numerical iteration. The optimization routines are able to optimize the ventilator settings individually or as a group.

Plan Evaluator. The plan evaluator provides a relative ranking of plans and their predicted consequences. This ranking is complicated by conflicting objectives, which are typical of medical decisions. In ventilator management, for example, increasing the percentage of oxygen in the breathing mixture

can improve the patient's oxygenation status, but high concentrations of oxygen have toxic effects. The optimal oxygen concentration represents a tradeoff between an improvement in oxygenation and an increase in oxygen toxicity. A multi-attribute value function is used to determine the optimal combination of objectives. The attributes of the value model are the proposed therapy plan (ventilator settings) and selected model predictions. A cost for each of the attributes is determined using a value function. These costs are weighted and summed to obtain the overall cost for a plan.

D. Relevant Publications

- 1) Fagan, L., Kunz, J., Feigenbaum, E, and Osborn, J. Adapting a rule-based system for a monitoring task, in *Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project*, B. Buchanan and E. Shortliffe (eds.). Reading, MA: Addison-Wesley Publishing Co., 1984.
- 2) Langlotz, C., Fagan, L., Tu, S., Sikic, B., and Shortliffe, E. A therapy planning architecture that combines decision theory and artificial intelligence techniques. *Computers and Biomedical Research* 20:279-303, 1987.
- 3) Rutledge, G., Thomsen, G., Beinlich, I., Farr, B. Kahn, M., Sheiner, L., and Fagan, L. VentPlan: An architecture for combining qualitative and quantitative computation. Report KSL-89-04, January 1989.

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A. Medical Collaborations and Program Dissemination via SUMEX

As described above, this project is a three-way collaboration between the Departments of Medicine, and Department of surgery at the VA Hospital, and U.C.S.F. As such we will need electronic mail and networking facilities. In addition, we imagine strong interactions with other projects around the world with similar research goals. We have already been contacted by research groups in Holland, Scotland, and Norway. In addition, similar research projects are underway at Yale, Berkeley, and Chicago. We expect that the networking facilities may allow us to share test cases, and possibly knowledge bases.

B. Sharing and Interaction with Other SUMEX-AIM Projects

The Yale project mentioned above is associated with Perry Miller's group. We also expect considerable interaction with the ONCOCIN and other parts of the Heuristic Programming Project at Stanford.

C. Critique of Resource Management

The SUMEX staff have been quite helpful in the support of the various machines that have been used in this project so far. We believe that the

current efforts of the SUMEX staff are quite appropriate for our research needs.

III. RESEARCH PLANS

Our basic research agenda is described above. The basic research issues underlying this project will extend for several years, leading to an implementation in the Veterans Administration Hospital in Palo Alto. This research will continue to need help assistance with local area networking, file service, and inter-network mail. We will need support for communications support within a project that is spread out over three geographical sites.

The SUMEX staff has been quite useful in providing support in other configurations of mainframe and workstations networked together. We anticipate that support for our unique collaborative arrangement will be equally superb.

IV.B. National AIM Projects

The following group of projects is formally approved for access to the AIM aliquot of the SUMEX-AIM resource. Their access is based on review by the AIM Advisory Group and approval by the AIM Executive Committee.

IV.B.1. INTERNIST-I/QMR Project

J. D. Myers, M.D.
University Professor Emeritus (Medicine)

Randolph A. Miller, M.D.
Associate Professor of Medicine
Chief, Section of Medical Informatics

University of Pittsburgh
1291 Scaife Hall
Pittsburgh, Pa., 15261

I. SUMMARY OF RESEARCH PROGRAM

A. Project rationale

The principal objective of this project is the development of a high-level computer diagnostic program in the broad field of internal medicine as an aid in the solution of complex and complicated diagnostic problems. To be effective, the program must be capable of multiple diagnoses (related or independent) in a given patient.

A major achievement of this research undertaking has been the design of a program called INTERNIST-I, along with an extensive medical knowledge base. This program has been used over the past decade to analyze many hundreds of difficult diagnostic problems in the field of internal medicine. These problem cases have included cases published in medical journals (particularly Case Records of the Massachusetts General Hospital, in the New England Journal of Medicine), CPCs, and unusual problems of patients in our Medical Center. In most instances, but by no means all, INTERNIST-I has performed at the level of the skilled internist, but the experience has highlighted several areas for improvement.

B. Medical Relevance and Collaboration

The program inherently has direct and substantial medical relevance.

The development of the QUICK MEDICAL REFERENCE (QMR) under the leadership of Dr. Randolph A. Miller has allowed us to distribute the INTERNIST-I knowledge base in a modified format to over twenty other academic medical institutions. The knowledge base can thereby be used as an "electronic textbook" in medical education at all levels -- by medical students, residents and fellows, and faculty and staff physicians. This distribution is continuing to expand.

The INTERNIST-I program has been used in recent years to develop patient management problems for the American College of Physician's Medical Knowledge Self-assessment Program.

C. Highlights of Research Progress

C.1 Accomplishments this past year

For the record, it should be noted that grant support for the QMR project has come solely from the CAMDAT Foundation of Farmington, Conn., from the Department of Medicine of the University of Pittsburgh, and from Dr. Miller's NLM RCDA grant, NLM RO1 grant and NLM UMLS contract.

In 1987, the University of Pittsburgh was named recipient of a National Library of Medicine Medical Informatics Training Grant Award.

The group of us (Myers, Miller and Masarie) together with assigned residents in internal medicine and fellows in medical informatics are continuing to expand the knowledge base and to incorporate the diagnostic consultative program into QMR. The computer program for the interrogative part of the diagnostic program is the main remaining task. An editor for the QMR knowledge base, as modified from the INTERNIST-I knowledge base, has been written from scratch in Turbo Pascal by Dr. Masarie. The entire QMR program can be accommodated in, maintained (particularly edited) and operated on individual IBM PC-AT computers.

Our group has incorporated into the QMR diagnostic consultant program modifications and embellishments of the INTERNIST-I knowledge base, and will continue to do so over the next year by adding "facets" of diseases or syndromes. This addition and modification is expected to improve the performance of the diagnostic consultant program.

The medical knowledge base has continued to grow both in the incorporation of new diseases and the modification of diseases already profiled so as to include recent advances in medical knowledge. Several dozen new diseases have been profiled during the past year. The current number of diseases in the QMR knowledge base is 597, and 4260 possible patient findings are included.

C.2 Research in progress

There are four major components to the continuation of this research project:

- 1) The enlargement, continued updating, refinement and testing of the extensive medical knowledge base required for the operation of INTERNIST-I and the QMR modification.
- 2) Institution of field trials of QMR on the clinical services in internal medicine at the Health Center of the University of Pittsburgh. This has been accomplished in a limited fashion, which began in 1987; a "computer-based diagnostic consultation service" has been made available to attending physicians and house staff on the medical services of our two main teaching hospitals. Institutional Review Board (IRB) approval was granted to the service before it was initiated.

- 3) Expansion of the clinical field trials to other university health centers which have expressed interest in working with the system.
- 4) Adaptation of the diagnostic program and data base of INTERNIST-I and the QMR modification to subserve educational purposes and the evaluation of clinical performance and competence.

Current activity is devoted mainly to the first two of these, namely, the continued development of the medical knowledge base, and the implementation of the improved diagnostic consulting program, and preliminary evaluation of the diagnostic consultation service.

D. List of relevant publications

- 1) Bankowitz RA, McNeil MA, Challinor SM, Parker RC, Kapoor WN, Miller RA. A Computer-Assisted Medical Diagnostic Consultation Service: Implementation and Prospective Evaluation of a Prototype. *Annals of Int Med.* 1989, 110(10):824-832.
- 2) Bankowitz RA, McNeil MA, Challinor SM. Effect of a Computer-Assisted General Medicine Diagnostic Consultation Service on Housestaff Diagnostic Strategy. *Proceedings of International Symposium on Medical Informatics and Education, Victoria, B.C., May 15-19, 1989, pp. 219-223.*
- 3) Giuse NB, Giuse DA, Miller RA. Learning by Doing: A Case Study in Medical Informatics. *Proceedings of the Sixth National Symposium on Computers in Medical Education. Omaha, Nebraska. March 28, 1989.*
- 4) Giuse NB, Giuse, DA, Miller RA. Computer assisted multi-center creation of medical knowledge bases. *Proceedings of the 12th Annual Symposium on Computer Applications in Medical Care. IEEE Press. pp. 583-90, November 1988.*
- 5) Giuse NB, Giuse DA, Miller RA. Medical knowledge base construction as a means of introducing medical students to medical informatics. *Proceedings of the International Symposium on Medical Informatics in Education. Victoria, B.C., May 15-19, 1989, pp. 228-232.*
- 6) Miller RA, et al. Preparing a Research Grant Proposal in Medical Informatics. *Comp Biomed Res.* 22(1):92-101, 1989.
- 7) Miller RA. Legal Issues Related to the Use of Medical Decision Support Systems. *International J Clin Monitoring and Computing.* 1989. (in press).
- 8) Berner ES, Brooks CM, Miller RA, Masarie FE Jr, Jackson JR. Evaluation Issues in the Development of Medical Decision Support Software. *Evaluation and the Health Professions.* Forthcoming 1990.
- 9) Lincoln M, Turner C, Hesse B, Miller RA. A Comparison of Clustered Knowledge Structures in Iliad and in Quick Medical Reference. *Proceedings of 12th Annual Symposium on Computer Applications in Medical Care. IEEE Press, pp. 131-135, November 1988.*

- 10) Miller RA, Masarie FE Jr. Use of the Quick Medical Reference (QMR) (R) Program as a Tool for Medical Education. Proceedings of the International Symposium on Medical Informatics and Education. Victoria, B.C. May 15-19, 1989, pp. 247-252.
- 11) Parker RC, Miller RA. Creation of a Knowledge Base Adequate for Simulating Patient Cases: Adding Deep Knowledge to the INTERNIST-1/QMR Knowledge Base. Proceedings the International Symposium on Medical Informatics and Education. Victoria, B.C. May 15-19, 1989, pp. 281-286.
- 12) McNeil M, Parker R, Bankowitz R, Challinor S. Correlates of internal medicine as a residency choice among students at the University of Pittsburgh. Society of General Internal Medicine Annual Meeting, 1989. (abstract)
- 13) Parker, S, Kroboth F, Parker R, Hanusa B, Kapoor W. Development of an easily completed and scored physician satisfaction questionnaire for use by both inpatients and outpatients. Society of General Internal Medicine Annual Meeting, 1989. (abstract)

E. Funding support

- 1) Diagnostic-Internist: A Computerized Medical Consultant
 Randolph A. Miller, M.D.
 Associate Professor of Medicine
 Chief, Section of Medical Informatics
 University of Pittsburgh Department of Medicine
 National Library of Medicine - Development Award Research Career
 National Institutes of Health
 5 KO4 LM00084
 09/30/85 - 09/29/86 - \$55,296
 09/30/86 - 09/29/87 - \$55,296
 09/30/87 - 09/29/88 - \$54,648
 09/30/88 - 09/29/89 - \$54,864
 Support recommended for 1 additional year ending 09/29/90. The Amount to be determined annually.
- 2) Developing INTERNIST-I Knowledge Base into a Resource
 Randolph A. Miller, M.D.
 Associate Professor of Medicine
 Chief, Section of Medical Informatics
 University of Pittsburgh Department of Medicine
 National Library of Medicine
 National Institutes of Health
 1 RO1 LM04622
 09/30/87 through 09/29/90
 09/30/87 - 09/29/88 - \$71,892
 09/30/88 - 09/29/89 - \$99,639
 09/30/89 - 09/29/90 - \$112,580

- 3) Pittsburgh Medical Information Sciences Training Program
 Randolph A. Miller, M.D.
 Associate Professor of Medicine
 Chief, Section of Medical Informatics
 University of Pittsburgh Department of Medicine
 National Library of Medicine
 National Institute of Health
 5 T15 LM07059
 07/01/87 through 06/30/92
 07/01/87 - 06/30/88 - \$153,454
 07/01/88 - 06/30/89 - \$221,511
 07/01/89 - 06/30/90 - \$265,930
 07/01/90 - 06/30/91 - \$278,092
 07/01/91 - 06/30/92 - \$276,596
- 4) Unified Medical Language System Support (UMLS)
 Randolph A. Miller, M.D.
 Associate Professor of Medicine
 Chief, Section of Medical Informatics
 University of Pittsburgh Department of Medicine
 National Library of Medicine
 N01-LM-8-3514
 06/30/88 through 06/29/91
 06/30/88 - 06/29/89 - \$133,617
 06/30/89 - 06/29/90 - \$129,573
 06/30/90 - 06/29/91 - \$188,824
- 5) Research Proposal for Implementation of Chinese Version
 of QMR
 Nunzia B. Giuse, M.D.
 Research Associate
 University of Pittsburgh Department of Medicine
 Section of Medical Informatics
 CAMDAT Foundation
 06/01/88 - 12/31/89
 \$16,500
- 6) Camdat Support of Quick Medical Reference (QMR)
 Nunzia B. Giuse, M.D.
 Research Associate
 University of Pittsburgh Department of Medicine
 Section of Medical Informatics
 CAMDAT Foundation
 08/01/88 - 12/31/89
 \$21,300

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A,B. Medical Collaborations and Program Dissemination Via SUMEX

INTERNIST-I and QMR remain in a stage of research and particularly development. As noted above, we are continuing to develop better computer programs to operate the diagnostic system, and the knowledge base cannot be used very effectively for collaborative purposes until it has reached a critical stage of completion. These factors have stifled collaboration via SUMEX up to this point and will continue to do so for the next year or two. In the meanwhile, through the SUMEX community there continues to be an exchange of information and states of progress. Such interactions particularly take place at the annual AIM Workshop.

C. Critique of Resource Management

SUMEX has been an excellent resource for the development of INTERNIST-I. Our large program is handled efficiently, effectively and accurately. The staff at SUMEX have been uniformly supportive, cooperative, and innovative in connection with our project's needs.

III. RESEARCH PLANS

A. Project Goals and Plans

Continued effort to complete the medical knowledge base in internal medicine will be pursued including the incorporation of newly described diseases and new or altered medical information on "old" diseases. The latter two activities have proven to be more formidable than originally conceived.

B. Justification and Requirements for Continued SUMEX Use

Our use of SUMEX has declined with the adaptation of our programs to the IBM PC-AT. Nevertheless, the excellent facilities of SUMEX are expected to be used for certain developmental work. It is intended for the present to keep INTERNIST-1 at SUMEX for comparative use as QMR is developed here. We will not need the DEC 2060 beyond its anticipated phase-out in early 1989, but will require access to its replacement for mailing purposes and to maintain contact with the national medical informatics community.

C. Needs and Plans for Other Computing Resources Beyond SUMEX-AIM

Our predictable needs in this area will be met by our recently acquired personal work stations.

IV.B.2. MENTOR Project

MENTOR Project — Medical Evaluation of Therapeutic Orders

Stuart M. Speedie, Ph.D.
School of Pharmacy
University of Maryland

Terrence F. Blaschke, M.D.
Department of Medicine
Division of Clinical Pharmacology
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

The goal of the MENTOR (Medical Evaluation of Therapeutic Orders) project is to design and develop an expert system for monitoring drug therapy for hospitalized patients that will provide appropriate advice to physicians concerning the existence and management of adverse drug reactions. The computer as a record-keeping device is becoming increasingly common in hospital-based health care, but much of its potential remains unrealized. Furthermore, this information is provided to the physician in the form of raw data which is often difficult to interpret. The wealth of raw data may effectively hide important information about the patient from the physician. This is particularly true with respect to adverse reactions to drugs which can only be detected by simultaneous examinations of several different types of data including drug data, laboratory tests and clinical signs.

In order to detect and appropriately manage adverse drug reactions, sophisticated medical knowledge and problem solving is required. Expert systems offer the possibility of embedding this expertise in a computer system. Such a system could automatically gather the appropriate information from existing record-keeping systems and continually monitor for the occurrence of adverse drug reactions. Based on a knowledge base of relevant data, it could analyze incoming data and inform physicians when adverse reactions are likely to occur or when they have occurred. The MENTOR project is an attempt to explore the problems associated with the development and implementation of such a system and to implement a prototype of a drug monitoring system in a hospital setting.

B. Medical Relevance and Collaboration

A number of independent studies have confirmed that the incidence of adverse reactions to drugs in hospitalized patients is significant and that they are for the most part preventable. Moreover, such statistics do not include instances of suboptimal drug therapy which may result in increased costs, extended length-of-stay, or ineffective therapy. Data in these areas are sparse, though medical care evaluations carried out as part of hospital quality assurance programs suggest that suboptimal therapy is common.

Other computer systems have been developed to influence physician decision making by monitoring patient data and providing feedback. However, most of these systems suffer from a significant structural shortcoming. This shortcoming involves the evaluation rules that are used to generate feedback. In all cases, these criteria consist of discrete, independent rules, yet medical decision making is a complex process in which many factors are interrelated. Thus, attempting to represent medical decision-making as a discrete set of independent rules, no matter how complex, is a task that can, at best, result in a first-order approximation of the process. This places an inherent limitation on the quality of feedback that can be provided. As a consequence it is extremely difficult to develop feedback that explicitly takes into account all information available on the patient. One might speculate that the lack of widespread acceptance of such systems may be due to the fact that their recommendations are often rejected by physicians. These systems must be made more valid if they are to enjoy widespread acceptance among physicians.

The MENTOR system is designed to address the significant problem of adverse drug reactions by means of a computer-based monitoring and feedback system to influence physician decision-making. It employs principles of artificial intelligence to create a more valid system for evaluating therapeutic decision-making.

The work in the MENTOR project is a collaboration between Dr. Blaschke at Stanford University, Dr. Speedie at the University of Maryland, and Dr. Charles Friedman at the University of North Carolina. Dr. Speedie provides the expertise in the area of artificial intelligence programming. Dr. Blaschke provides the medical expertise. Dr. Friedman contributes expertise in the area of physician feedback design and system impact evaluation. The blend of previous experience, medical knowledge, computer science knowledge and evaluation design expertise they represent is vital to the successful completion of the activities in the MENTOR project.

C. Highlights of Research Progress

The MENTOR project was initiated in December, 1983. The project has been funded by the National Center for Health Services Research since January 1, 1985. Initial effort focused on exploration of the problem of designing the MENTOR system. As of June 1, 1989, a working prototype system has been developed and is undergoing evaluation. The prototype consists of a Patient Data Base, an Inference Engine, an Advisory Module and a Medical Knowledge Base. The Medical Knowledge Base currently contains information related to aminoglycoside therapy, digoxin therapy, potassium supplementation, surgical prophylaxis, and microbiology lab reports. The system is currently implemented on a Xerox 1186 AI Workstation. Another version of the Patient Data Base has been developed for a VAXStation 3100 that is connected via an asynchronous line to the 1186 running the inference engine. The project has received additional funding from the National Center

for Health Services Research to install and evaluate the MENTOR system in a Veterans Administration Hospital. This effort began in June of 1988 and will continue for two additional years. The VA system will reside on an 1186 and a VAX Station II connected directly to the VA's Ethernet LAN, and accessing hospital data through the FILEMAN software.

E. Funding Support

Title: MENTOR: Monitoring Drug Therapy for Hospitalized Patients

Principal Investigators:

Terrence F. Blaschke, M.D.

Division of Clinical Pharmacology

Department of Medicine

Stanford University

Stuart M. Speedie, Ph.D.

School of Pharmacy

University of Maryland

Funding Agency: National Center for Health Services Research

Grant Identification Number: 1 R18 HS05263

Total Award: January 1, 1985 - May 31, 1990 \$1,091,750 Total

Direct Costs: Current Period: June 1, 1989 - May 31, 1990 \$289,961 (Total Direct Costs)

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A. Medical Collaborations and Program Dissemination via SUMEX

This project represents a collaboration between faculty at Stanford University Medical Center, the University of Maryland School of Pharmacy, and the University of North Carolina in exploring computer-based monitoring of drug therapy. SUMEX, through its communications capabilities, facilitates this collaboration of geographically separated project participants by providing electronic mail and file exchange between sites.

B. Sharing and Interactions with Other SUMEX-AIM Projects

Interactions with other SUMEX-AIM projects has been on an informal basis. Personal contacts have been made with individuals working on the ONCOCIN project concerning system development issues. Dr. Perry Miller has also been of assistance by providing software for advisory generation. Given the geographic separation of the investigators, the ability to exchange mail and programs via the SUMEX system as well as communicate with other SUMEX-AIM projects is vital to the success of the project.

C. Critique of Resource Management

To date, the resources of SUMEX have been fully adequate for the needs of this project. The staff have been most helpful with any problems we have had and we are quite satisfied with the current resource management.

III. RESEARCH PLANS

A. Project Goals and Plans

The MENTOR project has the following goals:

- 1) Implement a prototype computer system to continuously monitor patient drug therapy in a hospital setting. This will be an expert system that will use a modular, frame-oriented form of medical knowledge, a separate inference engine for applying the knowledge to specific situations, and automated collection of data from hospital information systems to produce therapeutic advisories.
- 2) Select a small number of important and frequently occurring medical settings (e.g., combination therapy with cardiac glycosides and diuretics) that can lead to therapeutic misadventures, construct a comprehensive medical knowledge base necessary to detect these situations using the information typically found in a computerized hospital information system and generate timely advisories intended to alter behavior and avoid preventable drug reactions.
- 3) Design and begin to implement an evaluation of the impact of the prototype MENTOR system on physicians' therapeutic decision-making as well as on outcome measures related to patient health and costs of care.

1988 will be spent on continued prototype development in six content areas, refinement of the inference mechanisms, and installation of the system at the Palo Alto Veterans Administration Hospital.

B. Justification and Requirements for Continued SUMEX Use

This project needs continued use of the SUMEX facilities for one primary reason. Access to SUMEX is necessary to support the collaborative efforts of geographically separated development teams at Stanford and the University of Maryland.

Furthermore, the MENTOR project is predicated on the access to the SUMEX resource free of charge over the next two years. Given the current restrictions on funding, the scope of the project would have to be greatly reduced if there were charges for use of SUMEX.

C. Needs and Plans for Other Computing Resources Beyond SUMEX-AIM

A major long-range goal of the MENTOR project is to implement this system on a independent hardware system of suitable architecture. It is recognized that the full monitoring system will require a large patient data base as well as a sizeable medical knowledge base and must operate on a close to real-time basis. Ultimately, the SUMEX facilities will not be suitable for these applications. Thus, we have transported the prototype system to a dedicated

hardware system that can fully support the the planned system and which can be integrated into a Hospital Information System. For this purpose a VAX 750 and three Xerox 1186 workstations have been acquired and our development efforts have been transferred to them.

D. Recommendations for Future Community and Resource Development

In the time we have been associated with SUMEX, we have been generally pleased with the facilities and services. However, it is clearly evident that the users' almost insatiable demands for CPU cycles and disk space cannot be met by a single central machine. The best strategy would appear to be one of emphasizing powerful workstations or relatively small, multi-user machines linked together in a nationwide network with SUMEX serving as the its central hub. This would give the individual users much more control over the resources available for their needs, yet at the same time allow for the communications among users that have been one of SUMEX's strong points.

For such a network to be successful, further work needs to be done in improving the network capabilities of SUMEX to encourage users at sites other than Stanford. Further work is also needed in the area of personal workstations to link them to such a network.

IV.C. Pilot Stanford Projects

Following are descriptions of the informal pilot projects currently using the Stanford portion of the SUMEX-AIM resource, pending funding, full review, and authorization.

IV.C.1. REFEREE Project

Principal Investigator: Bruce G. Buchanan, Ph.D.
Computer Science Department
Stanford University

Co-Principal Investigator: Byron W. Brown, Ph.D.
Department of Medicine
Stanford University

Associate Investigator: Daniel E. Feldman, Ph.D., M.D.
Department of Medicine
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

The goals of this project are related both to medical science and artificial intelligence: (a) use AI methods to allow the informed but non-expert reader of the medical literature to evaluate a randomized clinical trial, and (b) use the interpretation of the medical literature as a test problem for studies of knowledge acquisition and fusion of information from disparate sources. REFEREE and REVIEWER, a planned extension, will be used to evaluate the medical literature of clinical trials to determine the quality of a clinical trial, make judgments on the efficacy of the treatment proposed, and synthesize rules of clinical practice. The research is an initial step toward a more general goal - building computer systems to help the clinician and medical scientist read the medical literature more critically and more rapidly for use in making clinical decisions.

B. Medical Relevance

The explosive growth of the medical literature has created a severe information gap for the busy clinician. Most physicians can afford neither the time required to study all the pertinent journal articles in their field, nor the risk of ignoring potentially significant discoveries. The majority of clinicians, in fact, have little sophistication in epidemiology and statistics; they must nonetheless base their pragmatic decisions on a combination of clinical experience and published literature. The clinician's computerized assistant must ferret out useful maxims of clinical practice from the medical literature, pass judgment on the quality of medical reports, evaluate the efficacy of proposed treatments, and adjudicate the interpretation of conflicting and even contradictory studies.

C. Highlights of Progress

REFEREE presently encodes the methodological knowledge of a highly regarded biostatistician at Stanford (Dr. Bill Brown). The system allows the informed but non-expert reader of the medical literature to evaluate the credibility of a randomized clinical trial.

In the future, REFEREE and its extensions will alleviate the knowledge-acquisition bottleneck for an automated medical decision-maker: the program will help a reader to evaluate the quality of a clinical trial, judge the efficacy of the treatment proposed therein, and synthesize rules of clinical practice. For the present, however, the fusion of knowledge from disparate sources remains a problem in pure AI. The current effort of the REFEREE team is the appropriate representation of biostatistical knowledge in order to accomplish this set of tasks.

The REFEREE prototype is a consultant that evaluates the design and reporting of a single conclusion from randomized control trial for its credibility. It contains, in preliminary form, Professor Brown's expert knowledge of biostatistics. Given the assessments of the reader of various details, REFEREE synthesizes those judgments into a measure of credibility of the entire study. The reader may change his assessments in accordance with his uncertainty as to the judgments, and view graphically the resulting changes in REFEREE's measure.

The Knowledge Base:

Randomized controlled trials are used to test hypotheses regarding the effectiveness of various kinds of medical interventions. Dr. Brown classifies studies on the basis of three major attributes: the type of intervention tested (e.g., drug, surgery, health process change, etc.); the type of endpoint against which that intervention was tested (e.g. mortality, objective morbidity, subjective morbidity, etc.); and the type of conclusion drawn by the investigator/author on the basis of the research (e.g., that different treatments do or do not produce different outcomes, that a particular treatment is or is not cost-effective, etc.). Following this classificatory scheme, we decided to begin by producing a prototype REFEREE system that would help the reader to evaluate a single published conclusion concerning the effect of a given drug treatment on mortality.

Knowledge Acquisition:

Having defined the scope of the initial knowledge base, we turned to the problem of collecting the information from Dr. Brown for inclusion in the system, i.e., knowledge acquisition. This task generally involves a relatively long-term process of face-to-face information gathering during sessions between the expert and one or more knowledge engineers. Dr. Diana Forsythe has noted a parallel between the communicative and analytical tasks involved in knowledge acquisition and those undertaken in ethnographic research. For this reason, we included an anthropologist in the research team and make use of ethnographic techniques in order to maximize the efficiency and quality of the data collection process.

Dr. Lehmann and Dr. Forsythe have carried out a year of systematic interviews with Dr. Brown in order to begin the process of constructing and refining the knowledge base for the current REFEREE prototype. We have

combined a case-based approach that allows us actively to observe Dr. Brown as he reads papers, with semi-directed interviewing oriented toward understanding his terminology and category system. We find that these techniques work very well: Dr. Brown's interest in the knowledge acquisition process has been sustained, and indeed has increased over time as the system based on his expertise has evolved. He is clearly comfortable with this approach, and notes that it has actually afforded him additional insight into the way he interprets the literature.

Over the course of the project, we have altered our knowledge representation from that of rules to that of an *influence diagram*. This is an acyclic directed graph of propositions or variables connected by links, where the absence of a link indicates conditional independence of the two variables. This formalism has been used in decision analysis to enable experts to convey their knowledge of a domain, and, more recently, has been used in AI to represent that knowledge in expert systems. This shift in formalism significantly altered the knowledge acquisition process and the implementation of that knowledge in our program.

Based on information from our expert, we have taken *credibility* as the goal parameter of the present system. This goal is defined operationally by Dr. Brown as "my odds that the conclusion of interest would be replicated in an experiment based on the methods reported in the paper but without any of the flaws". In assessing *credibility*, for instance, Dr. Brown considers the blindedness of the randomization, the blindedness of the execution, the equivalence of the two groups at baseline, the equivalence in treatment of the two groups, the completeness of results reporting, and the propriety of the statistical analysis. We recognize that these variables are not all conditionally independent on *credibility*; work is in progress to assess as accurately as possible just what the conditional relationships are. Our use of influence diagrams has numerous advantages: the approach is acceptable to Dr. Brown, it is flexible, it can represent several aspects of the structure of the knowledge used by the expert, and the resultant data can be entered easily into the computer.

Inference in REFEREE:

REFEREE was originally built within EMYCIN, a backward-chaining rule-based AI environment developed from MYCIN at Stanford. This environment is ideally suited for ordered collection of evidence and a diagnosis of the goal state at the end of that process. The state of belief or knowledge in parameters not directly between the evidence and the goal state is irrelevant. Our present system focuses on maintaining consistency over the entire knowledge base as new evidence is incorporated into the system. The constraints implied by the new data and Dr. Brown's prior knowledge are propagated throughout the system by Judea Pearl's message-passing algorithm for belief networks. During the consultation with the program, questions are chosen by the user and answered at his or her discretion, and the state of belief in any parameter can be requested at any time. The odds of

replicating the study, then, can be viewed at any point during evidence collection.

There are a number of choices in representing our domain in an influence diagram. One is to view the goal of *credibility* as a proposition, the uncertainty in which is calculated by Pearl's algorithm. In this case, there are two choices: to view important design and execution factors as conditionally independent, given an assessment of the *credibility*, or to view them as causal of the goal measure. The program is currently implemented in the first topology, and we plan to test the second as well. A second representation is to view *credibility* as a measure of value, in which case the current knowledge base represents an objectives hierarchy, in the language of multi-attribute decision theory. We implemented REFEREE in David Klein's VERTUS system, following this paradigm, with moderate improvement in REFEREE's explanatory power.

A third representation is to reinterpret the task of REFEREE entirely, and to view it in the context of a physician's decision to treat or not to treat a patient with the intervention tested in the study under consideration. Dr. Lehmann is exploring this interpretation for his doctoral thesis.

The User Interface:

REFEREE was initially run entirely on the SUMEX resource. Mr. Chavez reimplemented the program on a stand-alone workstation, the Xerox 1186 in the KEE commercial expert system shell. The availability of bit-mapped screens made us more sensitive to issues of the user interface, but the shell could not deal easily with the uncertainty inherent in our domain. Mr. Chavez then ported the system to a Texas Instrument Explorer work-station, for which he designed an entirely new knowledge engineering shell which integrated EMYCIN and influence diagrams. It was apparent, however, that to accommodate the multiple interface needs of our potential user community, we needed a graphics environment that would allow frequent changes and customization. Thus, we turned to a final environment custom-made for influence-diagram-based expert systems. The KNET system, also developed by Mr. Chavez, separates the inferencing capabilities and graphical manipulation of the knowledge base into MPW Object Pascal from the textual part of the knowledge base and the the evidence collection in HyperCard. This system runs on the Macintosh II with 4 MB of RAM.

The program code is now entirely independent of the knowledge required for reading papers. REFEREE has a new interface that is intuitive and consistent. There is an innovative consultation mode in which questions are presented in free-format menus. The dialogues are mixed-initiative and of mixed levels, allowing the user such options as requesting more detailed questions or cutting off apparently fruitless lines of questioning. With the new REFEREE prototype, the user interacts with the machine using a mouse-pointing device. Finally, the screen enables the user to orient himself at all times, obviating the need for special commands to help the user

"navigate" through the knowledge base. Our expert recently provided the best indication of the usability of this new system. After only a brief introduction to the new machine and interface, he was able - for the first time - to run an entire consultation by himself.

Current Status:

At this point, REFEREE is a prototype that enables the clinician to read clinical trials more critically. A number of computational issues remain, such as the optimal representation of Dr. Brown's knowledge in our current formalism, and the decision-theoretic extensions. Furthermore, REFEREE represents only the first step in a larger research plan, the automation of knowledge acquisition (see section on Research Plans, below). Current work in the restricted domain of clinical trials will, we hope, illustrate general principles in the design of decision makers that gather expertise from written text and multiple knowledge sources.

D. Relevant Publications

- 1) Haggerty, J.: *REFEREE and RULECRITIC: Two prototypes for assessing the quality of a medical paper*. REPORT KSL-84-49. Master's Thesis, Stanford University, May 1984.
- 2) *Chavez, R. Martin and Cooper, G. F.: *KNET: Integration Hypermedia and Normative Bayesian Modeling*. Proceedings of the Fourth Workshop on Uncertainty in Artificial Intelligence, University of Minnesota, Minneapolis, Minnesota, Aug 19-21, 49-54, 1988.
- 3) *Lehmann, H. *Knowledge Acquisition for Probabilistic Expert Systems*. Proceedings of the Twelfth Symposium on Computer Applications in Medical Care, Washington, D.C., Nov 6-9, 73-77, 1988.
- 4) *Lehmann, H. *A Decision-Theoretic Model for Using Scientific Data* Submitted to the Fifth Uncertainty Workshop in Artificial Intelligence, 1989.

E. Funding Support

REFEREE currently receives only a small amount of funding. Most of the research is performed in time contributed by the researchers to this project.

Title: Knowledge-Based Systems Research

PI: Edward A. Feigenbaum

Agency: Defense Advanced Projects Research Agency

Grant identification number: N00039-86-0033

Total award period and amount: 10/1/85 - 9/30/88 \$4,130,230 (direct and indirect)

Current award period and amount: 10/1/87 - 9/30/88 \$1,467,300 (direct and indirect)

REFEREE component is \$27,706, or 1.9 % of grant total.

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A. Medical Collaborations

Dr. Brown and Dr. Feldman of the Stanford University School of Medicine are actively involved in the REFEREE project and are the primary domain experts and critics for this project.

C. Critique of Resource Management

The SUMEX computer resource and Lisp workstations have been very important for the work to date, and the SUMEX staff has continued to be very cooperative with the REFEREE project.

III. RESEARCH PLANS

A. Goals & Plans

The overall objective of the REFEREE project is to use recent Artificial Intelligence techniques to build a system that helps the informed but statistically non-expert reader to evaluate critically the medical literature on randomized controlled trials (RCT's). This system will contain and be able to apply dynamically the detailed specialized knowledge of Dr. Byron W. Brown, a biostatistician expert in the design and evaluation of randomized controlled trials. We have divided our overall objective into two goals:

- Goal 1 is the construction of an expert system to help readers (e.g., medical students, medical researchers, clinicians, journal editors, or editorial assistants) assess the credibility of a *single* conclusion drawn from a *single* journal report of a randomized controlled trial. We have already made substantial progress toward this goal with the development of the prototype REFEREE system.
- Goal 2 is the expansion of REFEREE to an expert system that can be used by a similar range of readers to facilitate the evaluation of *multiple* reports based on randomized controlled trials. This expanded system, to be known as the REVIEWER, will thus perform meta-analysis.

The task of extending and refining the prototype REFEREE system in order to achieve these goals can be characterized in terms of three dimensions:

- Making the system more accessible to a variety of people by improving the user interface, validating the system's performance with different types of users, and providing an explanatory capability
- Expanding the knowledge base by continuing the knowledge acquisition process to cover additional types of RCT's
- Improving the inference engine to ensure consistency of the knowledge base and to focus the consultation process on questions relevant to the situation and the individual user.

The specific steps that are planned for the enhancement of the REFEREE system include the following:

- Critique individual clinical trials according to the methodological quality of the trial;
- Measure the efficacy of treatment as demonstrated in a randomized control trial;
- Compare and contrast the credibility and efficacy of treatment reported by multiple journal articles; and
- Combine the *qualitative* techniques of heuristic reasoning and the *quantitative* methods of statistical meta-analysis to extract a consensus opinion from multiple knowledge sources.

In addition, plans for Goal 2, the REVIEWER system to analyze multiple RCT's and form a consensus judgment, include:

- Complete a review of the available literature on meta-analysis and augment the REFEREE prototype to produce estimators for meta-analysis and incorporate expert knowledge on the appropriateness of these methods.
- Add explicit and heuristic knowledge needed for the calculation of robust, non-parametric estimators of effect size.
- Construct a prototype of a system that builds categorical models in the domain of Bayesian meta-analysis, to perform autonomous investigations in the domain of statistical model-building. The REVIEWER will utilize expert knowledge in biostatistics to guide its search for meaningful models.
- Package the REVIEWER in a form suitable for use by physicians and their assistants.
- Verify the expertise of the REVIEWER system on a suite of papers drawn from clinical trials, similar to the validation of REFEREE above.

B. Justification for Continued SUMEX Use

The local area network maintained by the SUMEX staff is essential to the effective development and use of the REFEREE system on Lisp workstations. The connections to local and national computer networks such as ARPANET are important for sharing ideas and results with other medical researchers.

C. Need for other computing resources

REFEREE is currently implemented on the Macintosh II personal computers. We anticipate the need for at least two of these machines for transporting our system and developing new modes of interaction with both naive and experienced users.

IV.D. Pilot AIM Projects

Following is a description of the informal pilot projects currently using the AIM portion of the SUMEX-AIM resource, pending funding, full review, and authorization.

IV.D.1. The Pathfinder Project

Bharat Nathwani, M.D
Department of Pathology
University of Southern California

Lawrence M. Fagan, M.D., Ph.D
Department of Medicine
Stanford University

I. SUMMARY OF RESEARCH PROGRAM

A. Project Rationale

Our project addresses difficulties in the diagnosis of lymph node pathology. Several studies from cooperative oncology groups have documented that, while experts show agreement with one another, the diagnosis made by practicing pathologists may have to be changed by expert hematopathologists in as many as 50% of the cases. Precise diagnoses are crucial for the determination of optimal treatment. To make the knowledge and diagnostic reasoning capabilities of experts available to the practicing pathologist, we have been exploring issue of representation and inference with expert pathology knowledge. A computer-based diagnostic program called Pathfinder has been developed that is centered on the implementation of principles of probability and decision theory. The project is a collaborative effort of the University of Southern California and the Stanford University Medical Computer Science Group. The most recent version of the program provides diagnostic advice on over 70 common benign and malignant diseases of the lymph node based on over 100 histologic features. The design of the program, with special regard to the hypothetico-deductive reasoning architecture of the Pathfinder system, was influenced by the hypothetico-deductive architecture of the INTERNIST-1/CADUCEUS program developed on the SUMEX resource.

Pathfinder computer-science research is focused on the exploration and extension of formal techniques for decision making under uncertainty. Research foci have included (1) the assessment and representation of important probabilistic dependencies among morphologic features and diseases, (2) reasoning about the costs and benefits of alternative information acquisition strategies, (3) the acquisition and use of expert knowledge bases from multiple experts, (4) the customization of the system's reasoning and explanation behaviors to reflect the expertise of the user, and, (5) controlling the naturalness of complex formal reasoning techniques.

Toward the pragmatic goal of constructing a useful pathology teaching and decision-support system, Pathfinder investigators have sought to apply intelligent computation to substantially increase the quantity and quality of pathology knowledge available to pathologists. Important areas of this knowledge integration task involve ongoing research on the crisp definition important morphologic features and feature severities, the synthesis of

information from multiple experts, and the translation among multiple pathology classification schemes.

A group of expert pathologists from several centers in the U.S. have showed interest in the program and helped to provide the structure of the knowledge base for the Pathfinder system.

B. Medical Relevance and Collaboration

One of the most difficult areas in surgical pathology is the microscopic interpretation of lymph node biopsies. Most pathologists have difficulty in accurately classifying lymphomas. As mentioned above, several cooperative oncology group studies have documented that while experts show agreement with one another, the diagnosis rendered by a "local" pathologist may have to be changed by expert lymph node pathologists (expert hematopathologists) in as many as 50% of the cases.

The National Cancer Institute recognized this problem in 1968 and created the Lymphoma Task Force which is now identified as the Repository Center and the Pathology Panel for Lymphoma Clinical Studies. The main function of this expert panel of pathologists is to confirm the diagnosis of the "local" pathologists and to ensure that the pathologic diagnosis is made uniform from one center to another so that the comparative results of clinical therapeutic trials on lymphoma patients are valid. An expert panel approach is only a partial answer to this problem. The panel is useful in only a small percentage (3%) of cases; the Pathology Panel annually reviews only 1,000 cases whereas more than 30,000 new cases of lymphomas are reported each year. A panel approach to diagnosis is not practical and lymph node pathology cannot be routinely practiced in this manner.

We believe that practicing pathologists do not see enough case material to maintain a high level of diagnostic accuracy. The disparity between the experience of expert hematopathology teams and those in community hospitals is striking. An experienced hematopathology team may review thousands of cases per year. In contrast, in a community hospital, an average of only ten new cases of malignant lymphomas are diagnosed each year. Even in a university hospital, only approximately 100 new patients are diagnosed every year.

Because of the limited numbers of cases seen, pathologists may not be conversant with the differential diagnoses consistent with each of the histologic features of the lymph node; they may lack familiarity with the complete spectrum of the histologic findings associated with a wide range of diseases. In addition, pathologists may be unable to fully comprehend the conflicting concepts and terminology of the different classifications of non-Hodgkin's lymphomas, and may not be cognizant of the significance of the immunologic, cell kinetic, cytogenetic, and immunogenetic data associated with each of the subtypes of the non-Hodgkin's lymphomas.

In order to promote the accuracy of the knowledge base development we will have participants for multiple institutions collaborating on the project. Dr.

Nathwani will be joined by experts from Stanford (Dr Dorfman), St. Jude's Children's Research Center -- Memphis (Dr Berard) and City of Hope (Dr. Burke).

C. Highlights of Research Progress

C.1 Overview

Pathfinder research, has centered on the development of tractable methods for the acquisition, representation, and inference with probabilistic knowledge in pathology. Two M.D./Ph.D (Stanford Medical Information Science Program) students, David Heckerman and Eric Horvitz, designed and implemented the program and have played a central role in the direction of research on the project. In the past five years, the Pathfinder team has worked to (1) build a large consensus knowledge base of probabilistic inference, (2) to refine techniques of hypothetico-deductive reasoning, (3) to develop techniques for modulating the complexity of formal inference to enhance the clarity of reasoning and explanation, and (4) to begin formal evaluation of the performance of the system. Some of the Pathfinder research has stimulated other expert-systems research efforts centering on the re-examination of the construction of systems grounded in the principles of probability and decision theory.

C.2 History of Pathfinder System Implementation

Since the project's inception in September, 1983, we have constructed several versions of Pathfinder. The first several versions of the program were rule-based systems like MYCIN and ONCOCIN which were developed earlier by the Stanford group. These systems were implemented in the MRS logic theorem-proving language. We discovered early-on, however, that the large number of overlapping features in diseases of the lymph node would make a rule-based system cumbersome to implement. We next considered the construction of a hybrid system, consisting of a rule-based algorithm that would pass control to an INTERNIST-1-like scoring algorithm if it could not confirm the existence of classical sets of features. Later we, applied formal probabilistic representation and reasoning methods in a hypothetico-deductive reasoning framework. The original version of Pathfinder is written in the computer language MacLisp and runs on the SUMEX DEC-2060. This was transferred to Portable Standard Lisp (PSL) on the DEC-2060, and later transferred to PSL on the HP 9836 workstations. Two years ago, the Pathfinder team reimplemented the program in MPW Object Pascal on the Macintosh II. Much of the recent testing and refinement of the knowledge base has been carried out within the Macintosh II environment.

C.3 Pathfinder Knowledge Base

Initial versions of the Pathfinder knowledge base was constructed by Dr. Nathwani. During the early part of 1984, we organized two meetings of the entire team, including the pathology experts, to define the selection of

diseases to be included in the system, and the choice of features to be used in the scoring process. During the last three years, we have focused on methodologies for more accurately representing expert knowledge about the uncertain relationships between features and diseases in lymph-node pathology. Early versions of the Pathfinder knowledge base assumed independence between features used in diagnosis. However, knowledge-engineering sessions with the PI, who served as the chief hematopathology expert on the Pathfinder team, identified important probabilistic dependencies among features used in lymph node pathology. We have pursued the representation of the uncertain causal and associational relationships among features and diseases in lymph node pathology. We have found that attempting to move beyond the assumptions of conditional independence does not necessarily lead to an exponential growth in the tasks of knowledge acquisition, representation, and inference.

We have addressed the problem of probabilistic dependencies with a promising representation, developed in the decision science community, called belief networks. Although belief networks have been used as an alternative to decision trees for performing single analyses, there has been relatively little experience with the use of this representation in expert systems development. We pursued the use of belief networks because of the representation's soundness and expressiveness. With belief networks, probabilities are used to quantitate the beliefs about qualitative dependencies asserted by the expert. We found the belief network to be an intuitive and practical representation for building a large knowledge base. We have worked to enrich the basic belief network representation by developing a new language and associated operators for describing new types of conditional independence among findings. We found that a graphical knowledge-acquisition technique, called similarity-networks, could facilitate the knowledge acquisition process for building large, probability-based knowledge bases.

C.4 Simplification of Probabilistic Reasoning and Explanation

We have also focused on the problem of making complex information-theoretic inference understandable and explainable. We found that straightforward applications of decision-theoretic inference could lead to computer problem-solving behavior viewed as confusing or counterintuitive to users. Early, less-flexible versions of Pathfinder worked solely on the finest distinctions available in the system's representation. We found that users tended to work at higher levels of abstraction than did our straightforward decision-theoretic approach. Users also preferred to make specific transitions from one subproblem to another.

Knowledge acquisition with several pathologists unearthed alternative problem-solving control hierarchies that seemed to be used to segment a single complex diagnostic reasoning task (from the perspective of the decision-theoretic system) into a set of tasks at increasingly detailed levels of abstraction. These human-oriented abstraction strategies are useful for

allowing a pathologist to reason about groups of similar diseases rather than consider each disease as a separate entity. We have worked to acquire and apply alternative control strategies from trainees and experts. We worked to enhance the Pathfinder system to enable a user to probe a differential diagnosis from alternative perspectives. The current system allows a user to dynamically select alternative strategies for grouping the current differential list.

C.5 Evaluation of Pathfinder Performance

We applied a heuristic and decision-theoretic metric to perform a comparative analysis of the importance of enriching a conditional independence model with dependency knowledge. The study compared the performance of the system with that of the domain expert. In the evaluation study, a community pathologist used the Pathfinder system to analyze a set of difficult cases. Fifty-three cases were selected in sequence from a large library of referrals. As each case was entered into the system, probability distributions over disease hypotheses or differentials were generated by Pathfinder. In the next phase of the evaluation, the diagnostic accuracy of the distribution produced by Pathfinder was gauged by assigning it a score based on two metrics. We applied a heuristic scoring approach and a formal decision-theoretic approach. We found the two approaches to be complementary in their ability to identify components of system performance. The work showed a close correspondence between the behavior of the system and expert decision making.

C.6 SUMEX Usage

Although the SUMEX-AIM Resource was central in the initiation of the Pathfinder project, and in the prototyping of the early Pathfinder expert systems, the system not been used directly for development over the last three years. The resource has been used during this time for electronic mail and file archiving. Nevertheless, this SUMEX-AIM service has played a central role in communication among the participants on the Pathfinder project, especially for facilitating communication between the Stanford and USC Pathfinder research teams.

D. Publications Since January 1984

- 1) Horvitz, E. J., Heckerman, D. E., Nathwani, B. N. and Fagan, L. M.: "Diagnostic Strategies in the Hypothesis-directed Pathfinder System, Node Pathology." HPP Memo 84-13. Proceedings of the First Conference on Artificial Intelligence Applications, Denver, Colorado, Dec., 1984.
- 2) Heckerman, D. E., and Horvitz, E. J., "The Myth of Modularity in Rule-based Systems," in *Uncertainty in Artificial Intelligence*, Vol 2, J. Lemmer, L. Kanal, ed., North Holland, New York, 1987.

- 3) Horvitz, E. J., Heckerman, D. E., Nathwani, B. N. and Fagan, L. M.: "The Use of a Heuristic Problem-solving Hierarchy to Facilitate the Explanation of Hypothesis-directed Reasoning." KSL Memo 86-2 Proceedings of MedInfo, Washington D.C., October, 1986.
- 4) Horvitz, E. J., "Toward a Science of Expert Systems," Invited Paper, Computer Science and Statistics: Proceedings of the 18th Symposium on the Interface, American Statistical Association, March, 1986, pgs 45-52.
- 5) Heckerman, D. E., "An Axiomatic Framework for Belief Updates," in Uncertainty in Artificial Intelligence, Vol. 2, J. Lemmer, L. Kanal, ed., North Holland, New York, 1987.
- 6) Heckerman, D. E., and Horvitz, E. J., "The Myth of Modularity in Rule-based Systems," in Uncertainty in Artificial Intelligence, Vol 2, J. Lemmer, L. Kanal, ed., North Holland, New York, 1987.
- 7) Heckerman, D. E., and Horvitz, E. J., "On the expressiveness of rule-based systems for reasoning under uncertainty," Proceedings of the National Conference on Artificial Intelligence, Seattle, Washington, July, 1987.
- 8) Horvitz, E. J., Heckerman, D. E., Langlotz, C. P., "A framework for comparing alternative formalisms for plausible reasoning," Proceedings of the AAAI," August, 1986, Morgan Kaufman, Los Altos, CA, 1986.
- 9) Horvitz, E.J., Breese, J.S., Henrion, M., Decision Theory in Expert Systems and Artificial Intelligence, International Journal of Approximate Reasoning, Elsevier, N.Y. July, 1988.
- 10) Heckerman, D.E., An Evaluation of Three Scoring Schemes, Proceedings of the 4th AAAI Workshop on Uncertainty in Artificial Intelligence, Minneapolis, MN., (to appear August 1988).
- 11) Horvitz, E.J., A Multiattribute Utility Approach to Inference Understandability and Explanation, Tech. Report, KSL-28-87, Knowledge Systems Laboratory, Stanford, California, March, 1987.
- 12) Horvitz, E.J., "Reasoning About Beliefs and Actions Under Computational Resource Limitations," AAAI Workshop on Uncertainty in Artificial Intelligence, Seattle, Washington, 1987.
- 13) Horvitz, E.J., "Problem-solving Design: Reasoning About Computational Value, Resources, and Tradeoffs," Proceedings of the NASA Artificial Intelligence Forum. Palo Alto, California, November, 1987.
- 14) Nathwani, B.N., Horvitz, E.J., Heckerman, D.E., Lincoln, T., "Expert Systems and Interactive Videodiscs in Diagnostic Pathology: Augmenting the Multidisciplinary Approach," Human Pathology. In press.
- 15) Heckerman, E.J. Horvitz, B.N. Nathwani, "Toward Effective Normative Decision Systems: Update on the Pathfinder Project", Technical Report KSL-89-25, March, 1989. Knowledge Systems Laboratory, Stanford, CA; submitted to SCAMC-1989.

- 16) Horvitz, D.E. Heckerman, K. Ng, B.N. Nathwani, "Heuristic Abstraction in the Decision-Theoretic Pathfinder System," Technical Report KSL-89-24, March, 1989. Knowledge Systems Laboratory, Stanford, CA; submitted to SCAMC-1989.

E. Funding Support

Research Grant submitted to National Institutes of Health Grant
Title: "Computer-aided Diagnosis of Malignant Lymph Node Diseases"

Principal Investigator: Bharat Nathwani

Funding for three years from the National Library of Medicine

1 RO1 LM 04529 \$766,053 (direct and indirect)

Professional Staff Association, Los Angeles County Hospital, \$10,000

University of Southern California, Comprehensive Cancer Center, \$30,000

Project Socrates, Univ. of Southern Calif., Gift from IBM of IBM PC/XT.

II. INTERACTIONS WITH THE SUMEX-AIM RESOURCE

A. Medical Collaborations and Program Dissemination via SUMEX

Because our team of experts are in different parts of the country and the computer scientists are not located at the USC, we have made use of SUMEX for communication, demonstration of programs, and remote modification of the knowledge base.

B. Sharing and Interaction with Other SUMEX-AIM Projects

We have been in touch with other sites interested in Pathfinder research. As an example, the SUMEX pilot project, RXDX, designed to assist in the diagnosis of psychiatric disorders, is currently using a version of the Pathfinder program on the DEC-2060 for the development of early prototypes of future systems.

C. Critique of Resource Management

The SUMEX resource has provided an excellent basis for the development of a pilot project. The availability of a pre-existing facility with appropriate computer languages, communication facilities (especially the TELENET network), and document preparation facilities allowed us to make good progress in a short period of time. The management has been very useful in assisting with our needs during the start of this project.

III. RESEARCH PLANS

A. Project Goals and Plans

The current Pathfinder research grant will come to an end in Fall, 1989. The Pathfinder team is currently seeking a new grant to support a formal multicenter clinical trial to ascertain the efficacy of the use of system based on the Pathfinder knowledge base and inference techniques. We plan to

carry out a randomized trial of the use of the system with general pathologists. In addition to the statistical analysis of the efficacy of the system for providing assistance with the diagnostic subproblems of feature identification and integration, the group plans to study pathologists' attitudes on the use of computer-based decision support systems in the clinical environment.

B. Requirements for Continued SUMEX Use

We are currently dependent on the SUMEX computer for file storage and archival, and for communication. While the switch to workstations has lessened our requirements for computer time for the development of the algorithms, we will continue to need the SUMEX facility for the interaction with each of the research locations specified in our NIH proposal. An early version of the Pathfinder systems is stored on the SUMEX mainframe for use by non-Stanford users.

C. Requirements for Additional Computing Resources

Most of our computing resources will be met by the use of the Macintosh II workstations. However, we will continue to need additional file space on the SUMEX system for our continuing development and clinical trials work. We will also continue to require access to SUMEX for communication purposes, access to other programs, and for file storage and archiving.

Appendix A: Knowledge Systems Laboratory Brochure

ARTIFICIAL INTELLIGENCE RESEARCH IN THE KNOWLEDGE SYSTEMS LABORATORY

Stanford University
Department of Computer Science
Department of Medicine
March 1989

Introduction

The Knowledge Systems Laboratory (KSL) is an artificial intelligence (AI) research laboratory of approximately 100 people—faculty, staff, and students—within the Departments of Computer Science and Medicine at Stanford University. KSL is the name for the interdisciplinary AI research community that has evolved over the past two decades. Begun as the DENDRAL Project in 1965 and known as the Heuristic Programming Project from 1972 to 1984, the new organization reflects the diversity of the research now under way. The KSL is a modular laboratory, consisting of three collaborating yet distinct groups with different research themes:

- **The Heuristic Programming Project (HPP)**, Professor Edward A. Feigenbaum, scientific director (Department of Computer Science)—large, multi-use knowledge bases, blackboard systems, concurrent system architectures for AI, automated software design, expert systems for science and engineering. Executive director: Robert Englemore. Research scientists: Harold Brown, Bruce Delagi, Barbara Hayes-Roth, Yumi Iwasaki, Tom Gruber, Richard Keller, Hirotohi Maegawa, Penny Nii, and Kazuo Tanaka.
- **The Medical Computer Science (MCS) Group**, Associate Professor Edward H. Shortliffe, scientific director (Department of Medicine with courtesy appointment in Computer Science)—fundamental research and advanced biomedical applications in the area of AI and decision sciences; includes the Medical Information Sciences (MIS) program. Assistant Professor: Mark A. Musen. Associate Director: Lawrence M. Fagan. Research scientist: Gregory F. Cooper.
- **The Symbolic Systems Resources Group (SSRG)**, Thomas C. Rindfleisch, scientific director (joint appointment Departments of Computer Science and Medicine)—development of distributed computing environments for AI research and operation of KSL computing resources, including the SUMEX-AIM facility. SSRG Group Leaders: Richard Acuff, Christopher Lane, Nicholas Veizades, and William J. Yeager.

The KSL is guided by an Executive Committee consisting of the three sublaboratory directors and administrative managers. Tom Rindfleisch serves as overall KSL director (see Figure 1).

This brochure summarizes the goals and methodology of the KSL, its research and academic programs, its achievements, and the research environment of the laboratory.

Basic Research Goals and Methodology

Throughout a 24-year history, the KSL and its predecessors, DENDRAL and HPP, have concentrated on research in expert systems—that is, systems using symbolic reasoning and problem-solving processes that are based on extensive domain-specific knowledge. The KSL's approach has been to focus on applications that are themselves significant real-world problems (in domains such as science, medicine, engineering, and education), and that also expose key, underlying AI research issues. For the KSL, AI is largely an empirical science. Research problems are explored, not by examining strictly theoretical questions, but by designing, building, and experimenting with programs that serve to test underlying theories.

The basic research issues at the core of the KSL's interdisciplinary approach center on the computer representation and use of large amounts of domain-specific knowledge, both factual and heuristic (or judgmental). These questions have guided our work since the 1960's and are now of central importance in all of AI research:

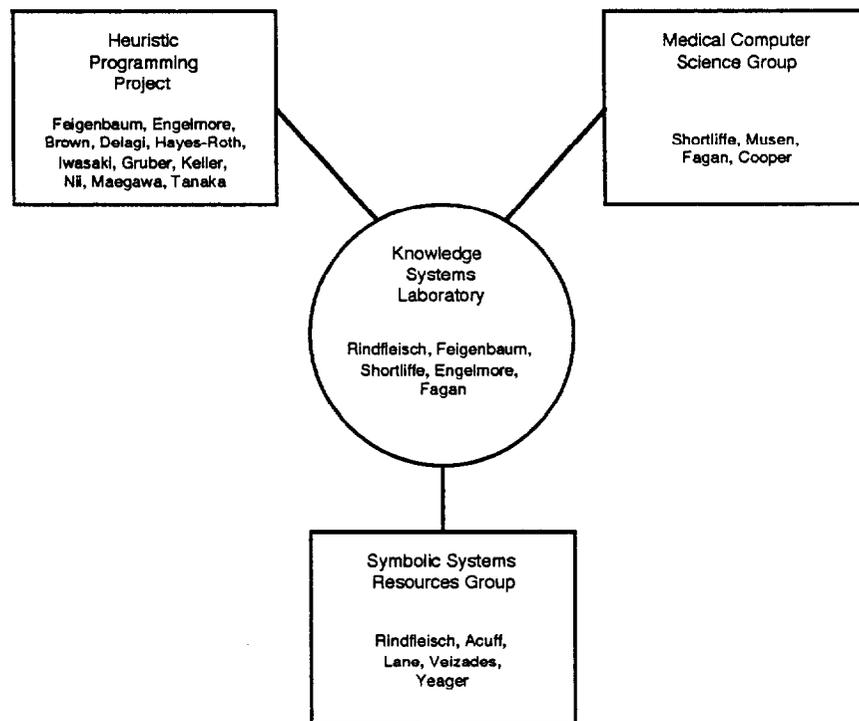


Figure 1 — Knowledge Systems Laboratory Organization

1. **Knowledge representation.** How can the knowledge necessary for complex problem solving be represented for its most effective use in automatic inference processes? Often, the knowledge obtained from experts is heuristic knowledge, gained from many years of experience. How can this knowledge, with its inherent vagueness and uncertainty, be represented and applied? How can knowledge be represented so that it can be used for many problem solving purposes? Can knowledge be abstracted for use in multiple ways?
2. **Knowledge acquisition.** How is knowledge acquired most efficiently—whether from human experts, from observed data, from experience, or by discovery? How can a program discover inconsistency and incompleteness in its knowledge base? How can knowledge be added without perturbing the established knowledge base unnecessarily?
3. **Use of knowledge.** By what inference methods can many sources of knowledge of diverse types be made to contribute jointly and efficiently toward solutions? How can knowledge be applied at the appropriate time and at the appropriate level of detail? How can existing knowledge be transformed so it is suitable for use by a specific application task?
4. **Explanation and tutoring.** How can the knowledge base and the line of reasoning used in solving a particular problem be explained to users? What constitutes a sufficient or an acceptable explanation for different classes of users?
5. **System tools and architectures.** What kinds of software tools and system architectures can be constructed to make it easier to implement expert programs with greater complexity and higher performance? What kinds of systems can serve as vehicles for the cumulation of knowledge of the field for the researchers? What architectural properties enable a system to function in real-time task environments?

Current Research Projects

The following is a summary of projects now under way within the three KSL research groups and gives the major goals of each project and lists the personnel (staff and Ph.D. candidates) directly involved. More complete information on individual projects can be obtained from the person indicated as the project contact. Inquiries should be addressed in care of:

Knowledge Systems Laboratory
Department of Computer Science
Stanford University
701 Welch Road, Building C
415-723-3444

The Heuristic Programming Project

- **Advanced Architectures Project**—Design a new generation of computer hardware architectures and problem solving frameworks to exploit concurrency in knowledge-based signal understanding systems.
Personnel: Edward A. Feigenbaum (contact), Nelleke Aiello, Harold Brown, Bruce Delagi (DEC), Robert Engelmores, Hirotohi Maegawa (Sony), Penny Nii, Sayuri Nishimura, James Rice, Nakul Saraiya.
- **Blackboard Architecture for Adaptive Intelligent Systems**—Design and develop a software architecture for systems that must reason about and interact with dynamic external entities in real time. Includes the Guardian project to develop a prototype system for real-time monitoring of surgical intensive care patients (see related VENTPLAN project under the Medical Computer Science Group).
Personnel: Barbara Hayes-Roth (contact), Richard Washington, Rattikorn Hewett, Adnan Darwiche, Michael Wolverton, Andrew Gans, Anthony Confrey, Luc Boureau, Anne Collinot, Iris Tommelein, Edward Chang, James Rice, Adam Seiver (Palo Alto VAMC).
- **Large, Multi-use Knowledge Base (LMKB) Project**—Develop an expert systems architecture capable of supporting multiple application tasks involving reasoning about engineered devices (e.g., device monitoring, diagnosis, redesign, assembly, instruction), using a large, common knowledge base of science and engineering principles underlying device design and operation.
Personnel: Edward Feigenbaum (contact), Richard Keller, Robert Engelmores, Yumi Iwasaki, Kazuo Tanaka (NTT), Tom Gruber.
- **Automated Software Design and Redesign**—Assist software designers in designing new systems via intelligent selection, modification, and construction from a library knowledge base of existing software modules.
Personnel: Penny Nii (contact), Cordell Green (Kestrel Institute), Nelleke Aiello, Raul Duran, Liam Peyton.

The Medical Computer Science Group

- **ONCOCIN**—Develop knowledge-based systems for the administration of complex medical treatment protocols such as those encountered in cancer chemotherapy.
Personnel: Ted Shortliffe (contact), Charlotte Jacobs (Oncology), Larry Fagan, David Combs, Robert Carlson, Christopher Lane, Rick Lenon, Mark Musen, Janice Rohn, Samson Tu, Cliff Wulfman, Andrew Zelenetz.
- **OPAL/PROTÉGÉ**—Develop graphics-based knowledge acquisition tools for clinical trials. OPAL developed out of the ONCOCIN project to provide a method for specifying cancer treatment experiments. The PROTÉGÉ program is capable of creating OPAL-like knowledge acquisition tools for various areas of medicine.
Personnel: Mark Musen (contact), David Combs.

- **Speech Input to Expert Systems**—Develop multi-modal interface to expert systems, concentrating on a connected speech input device. Primary application will be extension to the ONCOCIN graphical interface.
Personnel: Larry Fagan (contact), Bonnie Webber (University of Pennsylvania), Ted Shortliffe, Ed Feigenbaum (HPP), Ellen Isaacs (Psycholinguistics), Monica Rua, Clifford Wulfman, Christopher Lane, Janice Rohn.
- **Physician's Workstation**—Develop advanced integrated workstation suitable for providing decision support functions to clinicians in both inpatient and outpatient settings.
Personnel: Ted Shortliffe (contact), Tom Rindfleisch, Clifford Wulfman.
- **Qualitative and Quantitative Computation (VENTPLAN)**—Develop methods to combine qualitative and quantitative processing techniques in order to interpret and react to data gathered in time-varying application areas. The VENTPLAN system interprets data from the Intensive Care Unit, and suggests settings for mechanical ventilators (see related Guardian project in HPP).
Personnel: Larry Fagan (contact), Adam Seiver (Palo Alto Veterans Hospital), Lewis Sheiner (University of California, San Francisco), Ingo Beinlich, Brad Farr, Jeanette Polaschek, John Reed, Geoff Rutledge, George Thomsen, Samson Tu.
- **Decision-Theoretic Expert Systems**—Develop pragmatic methods of knowledge acquisition, inference, and explanation for medical expert systems based on decision theory.
Personnel: Greg Cooper (contact), Ted Shortliffe, Martin Chavez, David Heckerman, Edward Herskovits, Eric Horvitz, Harold Lehmann, Richard Lin, Blackford Middleton, Mike Shwe, Jaap Suermondt.

The Symbolic Systems Resources Group (SSRG)

- **SUMEX-AIM Resource**—Develop and operate a national computing resource for biomedical applications of artificial intelligence in medicine and for basic research in AI at KSL.
Personnel: Tom Rindfleisch (contact), Rich Acuff, Frank Gilmurray, Christopher Lane, Christopher Schmidt, Andrew Sweer, Bob Tucker, Nicholas Veizades, Bill Yeager.
- **AI Workstation and Network Systems**—Develop network-based computing environments for AI research on workstations including remote graphics and distributed computing.
Personnel: SSRG staff

Students and Special Degree Programs

Graduate students are an essential part of the research productivity of the KSL. Currently 30 students are working with our projects centered in Computer Science and another 24 students are working with the MCS/MIS programs in Medicine. Because of the highly interdisciplinary and experimental nature of KSL research, a special degree program, the *Medical Information Sciences (MIS)* program, was approved by Stanford University in 1982. It offers instruction and research opportunities leading to the M.S. or Ph.D. degree in medical information sciences. The program, directed by Ted Shortliffe and co-directed by Larry Fagan, is formally administered by the School of Medicine, but the curriculum and degree requirements are coordinated with the Dean of Graduate Studies and the Graduate Studies Committee of the University. The program reflects our local interest in the interconnections between computer science, artificial intelligence, and medical problems. Emphasis is placed on providing trainees with a broad conceptual overview of the field and with an ability to create new theoretical and practical innovations of clinical relevance. Of the 24 MIS students currently, 17 are working toward Ph.D. degrees, and 7 are working toward M.S. degrees.

Academic and Research Achievements

The primary products of our research are scientific publications on the basic research issues that motivate our work, computer software in the form of the expert systems and AI architectures we develop, and the students we graduate who continue AI research in other academic and industrial laboratories.

The KSL has averaged publishing more than 45 research papers per year in the AI literature, including journal articles, theses, proceedings articles, and working papers.¹ In addition, many talks and invited lectures are given annually. In the past few years, 12 major books have been published by KSL faculty, staff, and former students, and several more are in progress. Those recently published include:

- *Automated Generation of Model-Based Knowledge-Acquisition Tools*, Musen, Pitman, 1989.
- *Blackboard Systems*, Englemore and Morgan, eds., Addison-Wesley, 1988
- *The Rise of the Expert Company: How Visionary Companies are Using Artificial Intelligence to Achieve Higher Productivity and Profits*, Feigenbaum, McCorduck and Nii, Times Books, 1988.

¹ Copies of individual KSL publications may be obtained through the Stanford Department of Computer Science Publications Office. The full collection of KSL reports has been published in microfiche by COMTEX Scientific Corporation.

- *A Computational Model of Reasoning from the Clinical Literature*, Rennels, Lecture Notes in Medical Informatics, Volume 32, Springer-Verlag, 1987.
- *Heuristic Reasoning about Uncertainty: An AI Approach*, Cohen, Pitman, 1985.
- *Readings in Medical Artificial Intelligence: The First Decade*, Clancey and Shortliffe, Addison-Wesley, 1984.
- *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*, Buchanan and Shortliffe, Addison-Wesley, 1984.
- *The Fifth Generation: Artificial Intelligence and Japan's Computer Challenge to the World*, Feigenbaum and McCorduck, Addison-Wesley, 1983.
- *Building Expert Systems*, F. Hayes-Roth, Waterman, and Lenat, eds., Addison-Wesley, 1983.
- *System Aids in Constructing Consultation Programs: EMYCIN*, van Melle, UMI Research Press, 1982.
- *Knowledge-Based Systems in Artificial Intelligence: AM and TEIRESIAS*, Davis and Lenat, McGraw-Hill, 1982.
- *The Handbook of Artificial Intelligence*, Volume I, Barr and Feigenbaum, eds., 1981; Volume II, Barr and Feigenbaum, eds., 1982; Volume III, Cohen and Feigenbaum, eds., 1982; Kaufmann.
- *Applications of Artificial Intelligence for Organic Chemistry: The DENDRAL Project*, Lindsay, Buchanan, Feigenbaum, and Lederberg, McGraw-Hill, 1980.

Our laboratory has pioneered in the development and application of AI methods to produce high-performance knowledge-based programs. Programs have been developed in such diverse fields as analytical chemistry (DENDRAL), infectious disease diagnosis and treatment (MYCIN), cancer chemotherapy management (ONCOCIN), pulmonary function evaluation (PUFF), VLSI design (KBVLSI/PALLADIO), molecular biology (MOLGEN), parallel machine architecture simulation (CARE), and parallel problem solving (POLIGON). Some of our systems and tools (e.g., UNITS, EMYCIN, and AGE) have been adapted for commercial development and use in the AI industry.

Following our lead in work on biomedical applications of AI and the development of the SUMEX-AIM computing resource, a nationally recognized community of academic projects on AI in medicine has grown up.

KSL faculty, staff, and students have been recognized internationally for the quality of their work and for their continuing contributions to the field. KSL members participate extensively in professional organizations, government advisory committees, and journal editorial boards. They have held

managerial posts and conference chairmanships in both the American Association for Artificial Intelligence (AAAI) and the International Joint Conference on Artificial Intelligence (IJCAI).

Several KSL faculty and former students have received significant honors. In 1976, Ted Shortliffe received the Association of Computing Machinery Grace Murray Hopper award. In 1977, Doug Lenat was given the IJCAI Computers and Thought award, and in 1978, Ed Feigenbaum received the National Computer Conference Most Outstanding Technical Contribution award. In 1979 and 1981, Ted Shortliffe's book *Computer-Based Medical Consultation: MYCIN* was identified as the most frequently cited work in the IJCAI proceedings. In 1982, Doug Lenat won the Tioga prize for the best AAAI conference paper while Mike Genesereth received honorable mention. In 1983, Ted Shortliffe was named a Kaiser Foundation faculty scholar, and Tom Mitchell received the IJCAI Computers and Thought award. In 1984, Ed Feigenbaum was elected a fellow of the American Association for the Advancement of Science (AAAS), and he and Ted Shortliffe were elected fellows of the American College of Medical Informatics (ACMI). Larry Fagan was elected a fellow of ACMI in 1985. In 1986, Ed Feigenbaum was elected to the National Academy of Engineering and in 1987, Ted Shortliffe was elected to the Institute of Medicine of the National Academy of Sciences. The American Association for Medical Systems and Informatics Young Investigator Award for Research in Medical Knowledge Systems was presented to Glenn Rennels in 1988 and to Mark Musen in 1989.

KSL Research Environment

Funding—The KSL is supported solely by sponsored research and gift funds. We have had funding from many sources, including DARPA, NIH/NLM, ONR, NSF, NASA, and private foundations and industry. Of these, DARPA and NIH have been the most substantial and long-standing sources of support. All, however, have made complementary contributions to establishing an effective overall research environment that fosters interchanges at the intellectual and software levels and that provides the necessary physical computing resources for our work.

Computing Resources—Under the Symbolic Systems Resources Group, the KSL develops and operates its own computing resources tailored to the needs of its individual research projects. Current computing resources are a networked mixture of personal workstations, Lisp workstations, and central host computers and network utility servers, reflecting the evolving hardware technology available for AI research. Our central host is currently a Sun 4/280 running Sun Unix 4.0 (this is the core of the national SUMEX biomedical computing resource). It provides a central service for remote network access, electronic mail storage and routing, large-scale file storage, and printer spooling services. Increasingly, computing functions, such as electronic mail reading and composition, text processing, and information retrieval, are being moved to distributed user workstations. Our Lisp workstations include 34 Texas Instruments Explorers, 2 Symbolics 3600-

series machines, 3 SUN 3/75 workstations, and 4 NeXT machines. Much of the routine computing is done with 80 Apple Macintosh II computers, 15 of which have Texas Instruments microExplorer Lisp co-processor boards. Network printing, file storage, Internet gateway, and terminal interface services are provided by dedicated machines including a VAX 11/750, a SUN 3/180, and numerous special-purpose microprocessor systems. These facilities are integrated with other computer science resources at Stanford through an extensive Ethernet and to external resources through the ARPANET, TELENET, and the BARRNet (Bay Area Regional Research Network) link to the NSFNet. Funding for these resources comes principally from DARPA and NIH and hardware vendor gifts.

Appendix B: Lisp Performance Studies

Performance of Two Common Lisp Programs On Several Systems (Report KSL 89-02)

by Richard Acuff

Abstract

To assist in the evaluation of Lisp platforms for the Stanford University Knowledge Systems Laboratory, 22 Common Lisp implementations were benchmarked. Run time and compilation time data on two moderate-sized application programs are presented, along with data on the effect of compiler optimization levels and on the impact of display I/O on run time. For these Lisp benchmarks, several systems did not rank where we expected them based on speed ratings using other conventional measures. Also, the rankings of machines by Lisp speed differed for the two programs we tested. The data indicate that the performance of Lisp systems is very application dependent. Software environment should play at least as strong a role in machine selection as performance benchmarks.

1. Introduction

At Stanford University's Knowledge Systems Laboratory (KSL), a large amount of software is written in Lisp. Thus, the performance of Lisp systems is often crucial to the productivity of the lab. In order to assist us in understanding the performance of different Lisp systems, we have undertaken an informal survey of 22 Common Lisp implementations using two software packages developed in the KSL. The main goal of this survey was to understand the execution speed performance of systems that we might use in the KSL for research and development or dissemination of research results. Secondary goals were to evaluate the effect of compiler optimizer settings on execution speed and to evaluate the effect of reducing the amount of output on execution speed.

There have been a number of projects to measure the performance of Lisp systems. Gabriel's work [Gabriel 1985] is probably the best known, and is the origin of the so-called "Gabriel Benchmarks", a set of small test programs for measuring specific aspects of Lisp system performance. The Gabriel benchmarks are extremely valuable, for people trying to compare Lisp systems, if used knowledgeably. However, the aspects of a Lisp system stressed by a particular program are often difficult to determine so that it is usually best, where possible, to run that program on the systems in question rather than attempting to dissect the program and forecast its performance analytically. Also, with the advent of numerous implementations of Common Lisp [Steele 1984], we can now use much larger test programs without the bother and uncertainty of porting between dialects.

In this survey we have focused on execution speed which has long been an important criterion for comparing computer systems. The first comparison of

two systems solving the same problem (benchmarking) was probably made shortly after the creation of the second computer, and benchmarking has been a primary differentiator among computer systems ever since. However, execution speed benchmarks are only one aspect of the performance of systems, especially Lisp systems. Issues like programming and user environments, compatibility with other systems, the ability to handle "large" problems, and cost (hardware, software, and human) must also be considered, and, given a machine that is "fast enough", these other issues will almost always be the overriding factor.

Descriptions of the programs used in this evaluation are given in Section 2. A description of the methodology used in performing the tests is in Section 3, and information about the Lisp systems tested is in Section 4. Data on the execution speed of the test programs are presented in Section 5, followed by compilation speed data and a comparison between compilation speed and execution speed in Section 6. The effect of choosing various values for the SPEED and SAFETY options of the OPTIMIZE declaration on the BB1 system are discussed in Section 7. The effect of reducing the screen output of the SOAR benchmark is presented in Section 8. Details of the test procedures and descriptions of the systems tested are in the appendices.

2. Test Software

The software systems used in these tests were SOAR [Laird 1987] and the BB1 blackboard core [Hayes-Roth 1985 and Hayes-Roth 1988]. These test programs were chosen primarily because they are implemented in pure Common Lisp, making them extremely portable¹. Both are systems in daily use in the KSL but represent two distinct research directions in terms of program function and structure. These systems were initially developed in environments other than those tested, and no attempt was made to optimize their performance for any of these tests. Neither of these systems is an intensive user of numeric computation.

A copy of the Common Lisp source code used for these tests may be obtained from the author by sending U.S. Mail to "Richard Acuff, Stanford KSL, 701 Welch Road, Bldg. C, Stanford, CA 94305" or electronic mail to "acuff@SUMEX-AIM.Stanford.EDU".

¹ There were one or two small porting difficulties that were traced to problems in the test code which had to be fixed. For instance, many systems allow (INTERN "NAME" 'USER) where others require (INTERN "NAME" (FIND-PACKAGE "USER")). Also we were unable to get SOAR to work in either versions 1.0 or 1.1 of Allegro Common Lisp for the Mac II due to unexplained software hangs so it is omitted from SOAR-related charts.

2.1. SOAR

SOAR is a heuristic-search based general problem solving architecture developed by Paul Rosenbloom, *et. al.* See [Laird 1987] for more information on the SOAR system.

All test runs of SOAR were done solving an eight-puzzle problem in one of three modes: Mode A (simply solve the problem), Mode B (solve the problem while "chunking" or "learning"), and Mode C (solve the problem after having "learned" in Mode B).

An "eight puzzle" is a common children's game with 8 tiles, numbered 1 to 8, on a 3 by 3 grid such that a tile adjacent to the empty place can be pushed into it. "Solving the eight-puzzle problem" consists of producing a series of tile moves such that, from a given arbitrary starting configuration, the eight puzzle ends up with all the tiles in numerical order, reading from the upper left around the puzzle clockwise, with the empty place in the middle.

The version of SOAR used was 4.4.4, dated April 19, 1987. It consists of 1 large LISP source file and 2 small SOAR files containing productions for solving the eight-puzzle problem. The LISP source is 10,661 lines (280,050 characters) of lightly commented code.

2.2. BB1

BB1 is a blackboard-based problem solving architecture developed by Barbara Hayes-Roth. For more information on the BB1 blackboard core, see [Hayes-Roth 1985]. For further information on BB1, see [Hayes-Roth 1988]. All references to BB1 in this document refer only to the "core" blackboard parts of the system and do not include any other layers of the problem solving architecture or the user interface, as these components are not in pure Common Lisp. All test runs of BB1 went through three cycles of adding 10 items to the blackboard, accessing those 10 items, and then deleting them.

The version of BB1 used was 1.2. The LISP source used consists of 10 files ranging from 36 lines (814 characters) to 3,396 lines (107,528 characters) of lightly commented code, with a total of 8,722 lines (295,199 characters) of code.

3. Methodology

All the tests were performed in as near to a "normal" working environment as could be achieved. We tried to duplicate the working conditions that a researcher would likely have both in hardware and software. Where possible we selected test machines configured with the amount of memory, amount and type of disk, type of display, etc. that a typical developer would purchase and use. We ran the software in a way that a developer using the system would probably use it. Thus, if it was normal to run with garbage collection enabled, under a window system, within an editor, or in a multi-programming environment, then that was done. For instance, Sun machines were tested under *SunView* with a couple of *perfmeters* running. The HP

machine was tested while running in *GnuEmacs* on *X.10*. MIT-style Lisp machines were run with all networking and other background processing on, and no special process priority. No expert tuning or system configuration was done beyond what the tester could do by reading over the user documentation. All systems were tested in single-user mode, which is the way those tested are normally used for Lisp work.

We feel that although this methodology results in less repeatable and less explainable results, it gives a good approximation to what the end user will experience. Where time allowed, multiple runs were made to ensure accurate readings. Unfortunately the collection of the raw data (i.e. arranging for machine access and making the timed runs) proved to be an extremely time consuming process, taking a day or more for some of the systems, so the information in this report was collected over a long period of time (October, 1987 to January 1989) and some of the data may be dated by now.

The procedures used for running the tests are fully described in Appendix B. The `TIME` macro was used to collect timing information. Most times were recorded to the nearest second. When reported by the `TIME` macro, some extra information, usually relating to paging, memory management, "kernel" time, etc., were recorded, but are not analyzed here. If several runs were made, only the best number is reported herein for the sake of brevity. Wherever possible, source files were stored on local disks (for the Sun 3/75 systems the files were on a Sun 3/180 NFS server on the same subnet).

4. Systems Under Test

The systems that we tested were chosen based on their availability to the testers as well as their suspected usefulness in future KSL programming efforts. All of the systems tested were workstations, as we were not able to obtain access to mainframe systems. It is also the case that workstations, with their bit-mapped displays and dedicated processors, currently provide the best Lisp development environments, in our opinion, and thus were more interesting to us.

A mnemonic code is used for each of the 22 systems. Usually the code is the model of the machine except where there is more than one Lisp for a machine (as in the case of the Sun 3/75) in which case a letter is prefixed to indicate the Lisp being used. Table 1 gives a mapping between codes and machine types. See Appendix A for detailed descriptions of system configurations.

5. Execution Speed

Most of the tables and charts in this report refer to elapsed-times (wall-clock time) in seconds. Most of the tables and charts have the system types ordered according to what seems to be the most interesting comparison. We have attempted to group systems of allegedly comparable performance (according to our perception formed from talking to vendor representatives, talking to other users, reading reports, etc.)

<u>Code</u>	<u>Test Date</u>	<u>System Type</u>
3/260	Summer 1988	Sun 3/260 with Lucid Lisp ¹
3/60	Summer 1988	Sun 3/60 with Lucid Lisp
386	Spring 1988	Compaq 386 with Lucid Lisp
386T	Spring 1988	Compaq 386 portable with Lucid Lisp
4/260	Summer 1988	Sun 4/260 with Lucid Lisp
4/280	Winter 1988	Sun 4/280 with Lucid Lisp
DEC-II	Fall 1987	DEC MicroVax II with VaxLisp
DEC-III	Fall 1987	DEC MicroVax III with VaxLisp
E-3/75	Fall 1987	Sun 3/75 with Franz Extended Common Lisp
EXP1	November 1988	Texas Instruments Explorer I
EXP2	November 1988	Texas Instruments Explorer II
EXP2+	November 1988	Texas Instruments Explorer II Plus
F-4/280	January 1989	Sun 4/280 with Franz Allegro Common Lisp
HP	Fall 1987	Hewlett Packard 9000/350
K-3/75	Fall 1987	Sun 3/75 with Kyoto Common Lisp
L-3/75	Summer 1988	Sun 3/75 with Lucid Lisp
Mac2	Spring 1988	Apple Macintosh II with Allegro Common Lisp
Maci	December 1988	Symbolics MacIvory
mX	November 1988	Texas Instruments microExplorer
RT	Spring 1988	IBM RT/APC with Lucid Lisp
Sym	Winter 1988	Symbolics 3645
XCL	Winter 1988	Xerox 1186

Table 1: Mapping between codes and system types

It is worth noting that on almost all of the systems tested, virtual memory paging was a negligible part of the overall run time for the tests. Nor was it a very significant factor during compilation. In general, we do not expect this to be true for most production systems. Indeed, we would not be surprised if paging time were a major component of overall run time for most systems.

5.1. BB1

The data for the run times of the BB1² tests are given in Table 2. Figure 1 shows the data graphically.

-
- ¹ The Lucid and Franz Extended Common Lisp products tested are versions prior to multiprogramming within the Lisp and prior to the inclusion of generation-based scavenging garbage collection in those systems. The Allegro Common Lisp was not tested with multiprogramming enabled.
 - ² These times are for default settings of the SPEED and SAFETY optimization qualities discussed in Section 7.

<u>Code</u>	<u>Run Time</u>	<u>Code</u>	<u>Run Time</u>
Exp2	27	RT	75
Exp2+	17	DEC-III	63
4/260	56	Exp1	87
4/280	34	3/60	73
-4/280	56	L-3/75	90
386	47	E-3/75	211
386T	54	K-3/75	96
mX	33	HP	115
Mac1	129	DEC-II	207
Sym	111	XCL	559
3/260	62	Mac2	254

Table 2: Run times for BB1

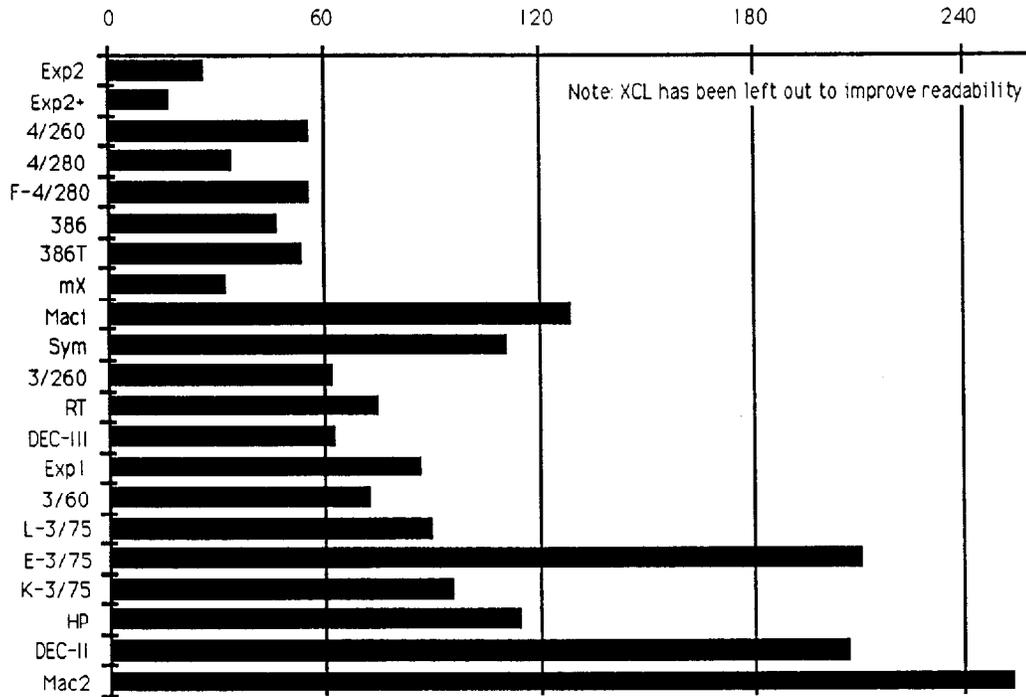


Figure 1: BB1 Run (sec)

Systems that are marketed as comparable generally came out close to each other with the following notable exceptions:

- There was a significant difference between the 4/280 and the 4/260. Even though the 4/260 had more memory, similar disk, more tuning effort, and was tried with several later versions of Lisp it was consistently slower than the 4/280 tested earlier. We are at a loss to explain this discrepancy.

It is also worth noting that, except for VaxLisp, Lucid Lisp seemed the most difficult to tailor to a particular machine when it was being installed.

- The DEC machines seem to be poor at running Lisp even though they are usually thought of as competitive when running FORTRAN or C.
- The microExplorer (mX) did better than expected probably because its weak point, paging, was not stressed by this test.
- The much older Franz Lisp (E-3/75) did relatively poorly compared to Lucid Lisp on the 3/75, but the newer version on the Sun 4 did well relative to the somewhat older Lucid lisp on the Sun 4.
- XCL was over twice as slow as the nearest competitor.
- For unknown reasons the Symbolics machines were slower than expected. The MacIvory was a bit over 4 times slower than the microExplorer and the 3645 was slower than the Explorer I.

5.2. SOAR

The data for the SOAR run tests are given in Table 3 and presented graphically in Figure 2. The figures are for the sum of the A, B, and C modes¹.

Once again most systems fit where expected with the following notes:

- The Lucid Sun 4's are somewhat faster than the TI Explorer II for the SOAR test whereas the opposite was true for the BB1 test.
- XCL and DEC-II were over twice as slow as the nearest other system.

<u>Code</u>	<u>Run Time</u>	<u>Code</u>	<u>Run Time</u>
Exp2	94	RT	177
Exp2+	62	DEC-III	454
4/260	58	Exp1	369
4/280	82	3/60	187
-4/280	120	L-3/75	278
386	126	E-3/75	484
386T	151	K-3/75	697
mX	154	HP	219
Maci	339	DEC-II	1851
Sym	193	XCL	1519
3/260	154	Mac2	No data (see footnote 1)

Table 3: Aggregate Run Times for SOAR

¹ The A and C mode figures are for the "no trace" configuration as described in Section 8.

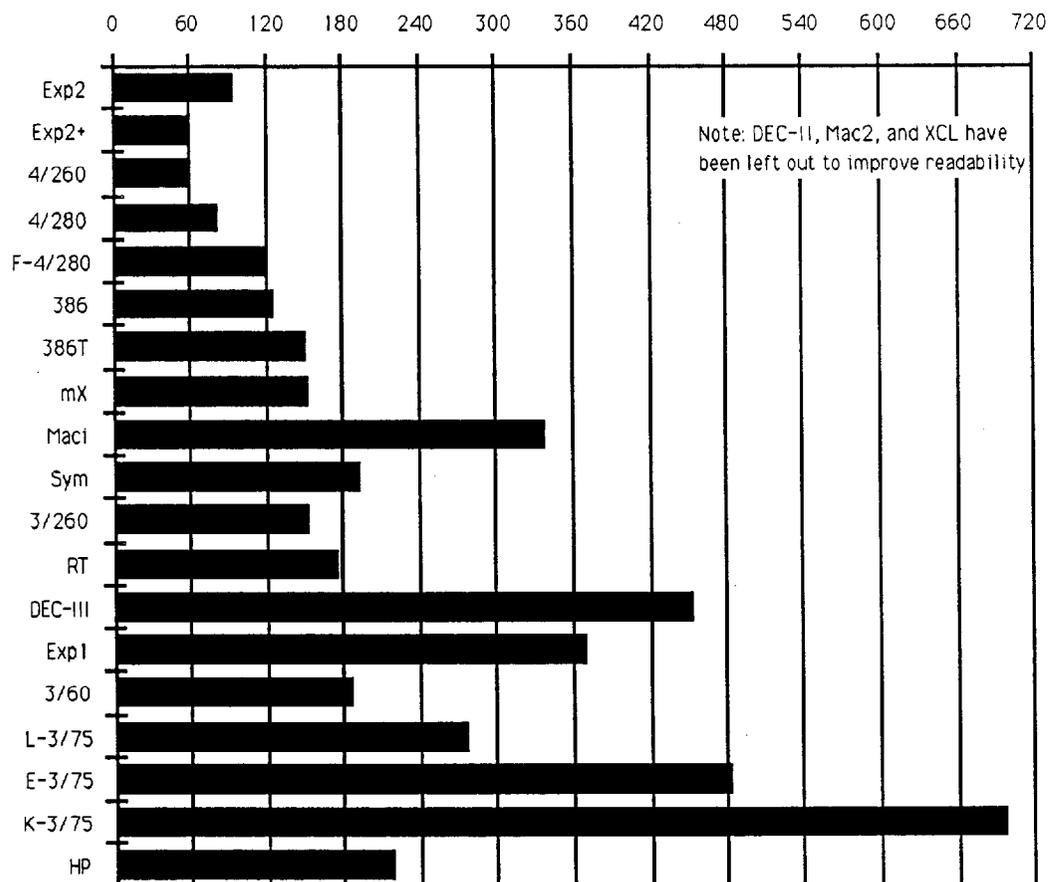


Figure 2: Sum of SOAR run times (sec)

5.3. Normalized Run Times

A given machine, call it *A*, may have run the SOAR test faster than another machine, *B*, while *B* was faster for BB1. Figure 3 depicts this difference. For both BB1 and SOAR the run times have been normalized by dividing the run time by the average of the run times for all the machines, leaving out DEC-II, Mac2, and XCL to improve readability.

Lucid Lisp seemed to perform relatively better with SOAR than with BB1 in all cases, while VaxLisp and, to a much lesser extent, the dedicated Lisp machines, seemed to do better with BB1.

There are many possible explanations for these variations, but trying to analyze each of them was well beyond the scope of this study. The reasons are most likely a result of differences among implementations in the efficiency of various operations, some of which are used by SOAR but not by BB1 and *vice versa*. For instance, SOAR might make heavy use of hashing

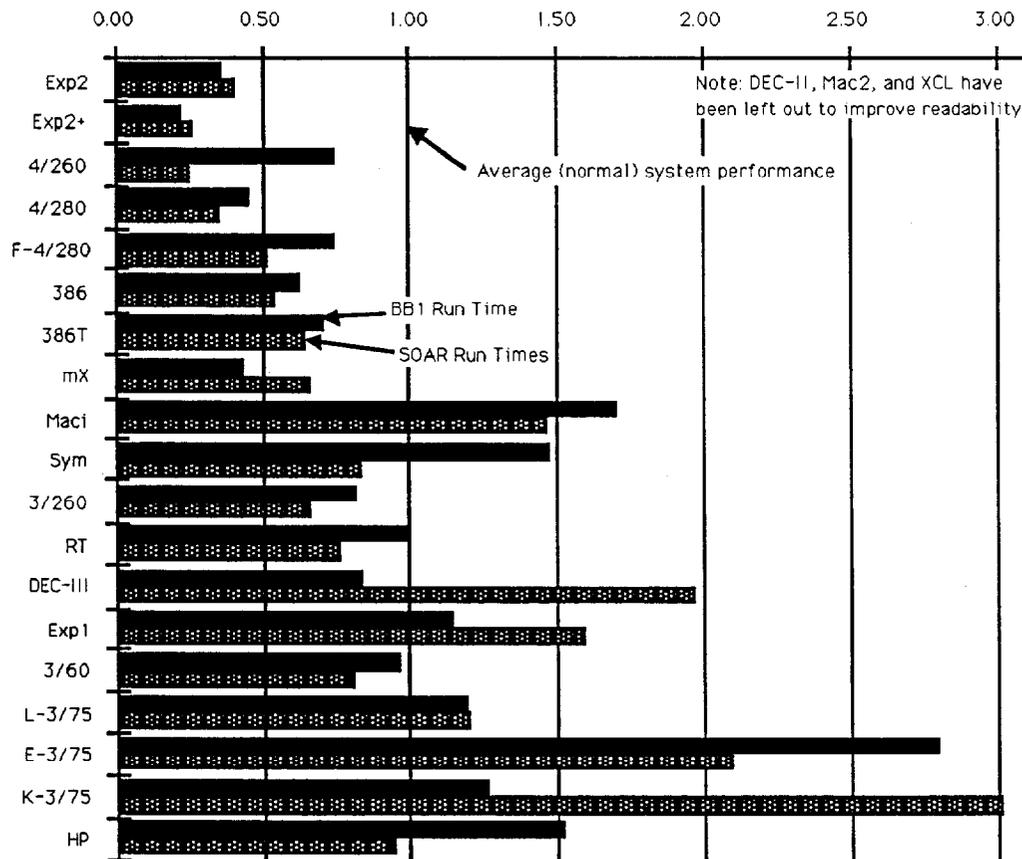


Figure 3: Normalized Run Times (time/average_time)

while BB1 makes heavy use of list primitives, or one system might include a large number of SETQ operations while the other might be more applicative in nature. The developers of SOAR and BB1 do not currently have information on the aspects of the Lisp systems stressed by their software.

6. Compilation Speed

Developers and researchers must worry about how fast their programs compile as well as how fast they run. SOAR and BB1 compilation times are given in Table 4 and Figure 4.

Figures 5 compares run time with compile time. The ratio of compilation time to run time is shown. A system with a high rating spends relatively more time compiling than running. The absolute value of these numbers have little meaning. They are only useful for comparing systems.

<u>Code</u>	<u>SOAR</u>	<u>BB1</u>	<u>Code</u>	<u>SOAR</u>	<u>BB1</u>
Exp2	132	89	RT	574	586
Exp2+	78	76	DEC-III	423	633
4/260	307	324	Exp1	520	327
4/280	523	482	3/60	569	551
F-4/280	535	264	L-3/75	1040	919
386	386	355	E-3/75	450	444
386T	479	416	K-3/75	1365	1234
mX	152	186	HP	237	235
Maci	906	950	DEC-II	1227	1774
Sym	252	257	XCL	1800	1927
3/260	687	540	Mac2	0	349

Table 4: Compilation Times

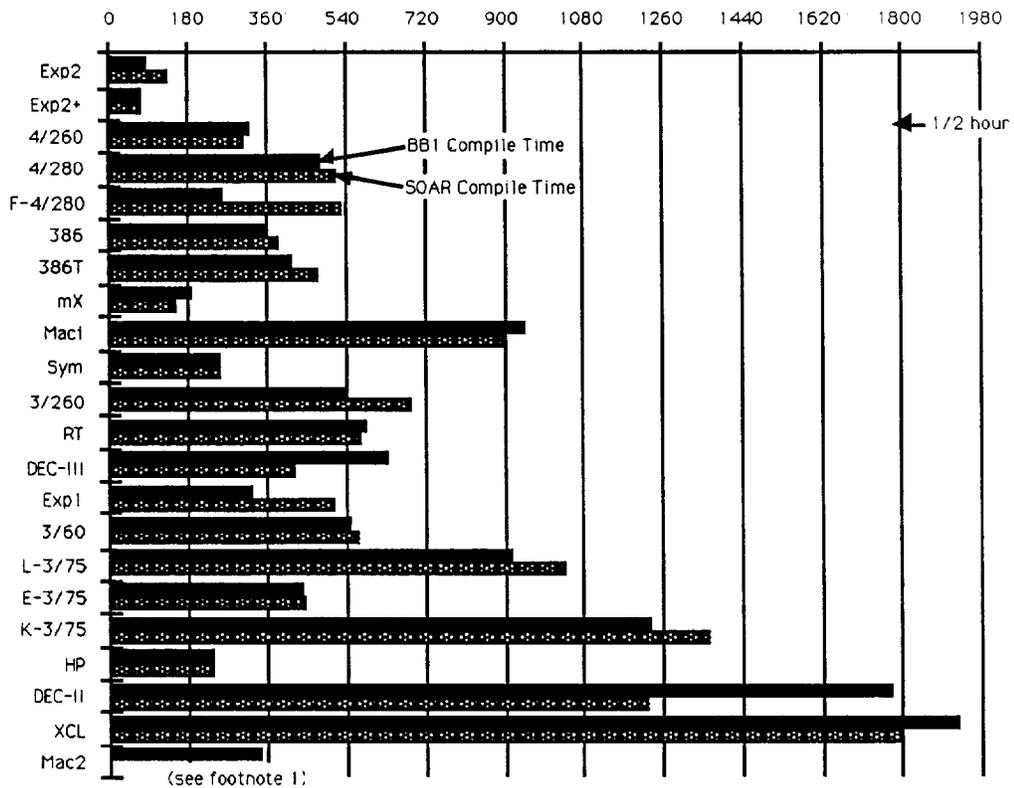


Figure 4: Compilation Time (sec)

As one might expect, the specially microprogrammed Lisp machines had relatively fast compilers. Some machines with run times slower than predicted spent relatively less time compiling. For example, the VaxLisp compiler was relatively fast, but generated very slow code. The Lucid compiler seemed to take a long time but generated fast code. The Allegro

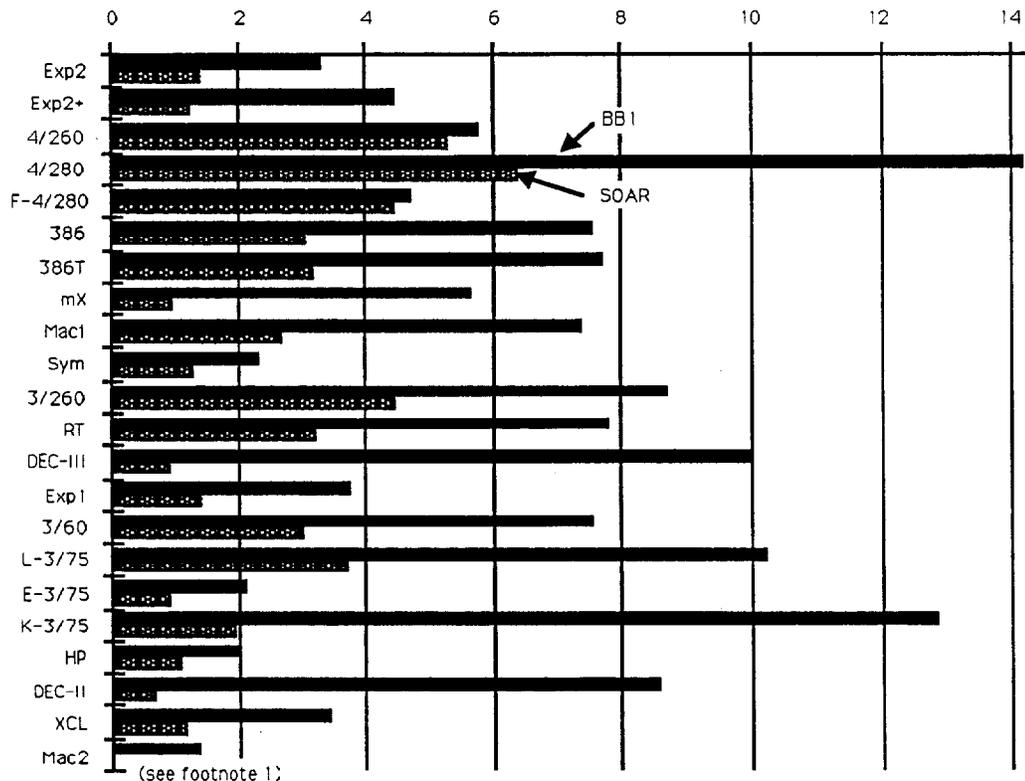


Figure 5: Relative Performance of Compiler
(Compile_Time/Run_Time)

Common Lisp for the Mac II took little time but still somehow generated impressively fast code for BB1.

7. Effect of OPTIMIZE Settings on BB1

The OPTIMIZE declaration is a way of controlling the behavior of a Common Lisp compiler. Two of the most significant qualities thus controlled are SPEED and SAFETY. Each of these can be set to an integer from 0 to 3. A high setting for SPEED tells the compiler that fast running code is desired, which typically enables various optimizations. The Common Lisp specification doesn't require any optimizations or even that they necessarily be controlled by this setting, but many current implementations switch on optimizers such as dead code eliminators, tail and mutual recursion eliminators, fancy register allocators, and facilities to take advantage of type declarations. The SAFETY quality is somewhat less well understood. It has little to do with the "safety" of the program since a correct Common Lisp program is still required to run correctly if SAFETY is low, but it has an impact on the debuggability of the program. A high SPEED and low SAFETY may allow, for instance, disabling number-of-arguments checking to allow

faster function calls on some architectures, or type checking on system functions (such as CAR or SETQ) might be disabled. Kyoto Common Lisp (KCL) goes so far as to "hardwire" function calls such that if FOO calls BAR and FOO is compiled then if BAR is later redefined and FOO isn't, FOO will continue to call the old version of BAR, thereby destroying much of the flexibility of the Lisp.

We chose 4 settings of SPEED and SAFETY to study:

1. The default setting that the Lisp system has when it is initialized. This is what most people use.
2. SPEED 3, SAFETY 0 (written (3, 0) below) which should generate the fastest code.
3. SPEED 0, SAFETY 3 (written (0, 3) below) which should generate slow but very debuggable code, since the compiler should have done very few, if any, optimizations.
4. SPEED 3, SAFETY 2 (written (3, 2) below) which should generate optimized code while retaining "sanity checks".

The BB1 system used in these tests has very few declarations and does little numerical work. Both of these attributes seem common among most Common Lisp programs we use.

<u>Code</u>	<u>Default</u>	<u>(3, 0)</u>	<u>(0, 3)</u>	<u>(3, 2)</u>
Exp2	27	25	27	25
Exp2+	17	17	18	18
4/260	56	46	47	46
4/280	34	34	48	34
F-4/280	56	56	56	54
386	47	47	52	47
386T	54	54	60	54
mX	33	34	34	30
Maci	129	129	130	130
Sym	111	109	110	111
3/260	62	62	69	62
RT	75	76	77	75
DEC-III	63	60	71	70
Exp1	87	87	90	83
3/60	73	72	76	72
L-3/75	90	90	127	90
E-3/75	211	215	206	206
K-3/75	96	165	147	88
HP	115	113	141	118
DEC-II	207	206	231	236
XCL	559	543	559	556
Mac2	254	258	261	259

Table 5: BB1 Run Times for Various OPTIMIZE Settings

Table 5 and Figure 6 give the results for running BB1 with the four OPTIMIZE settings. Figure 7 shows the compilation times for the various OPTIMIZE settings.

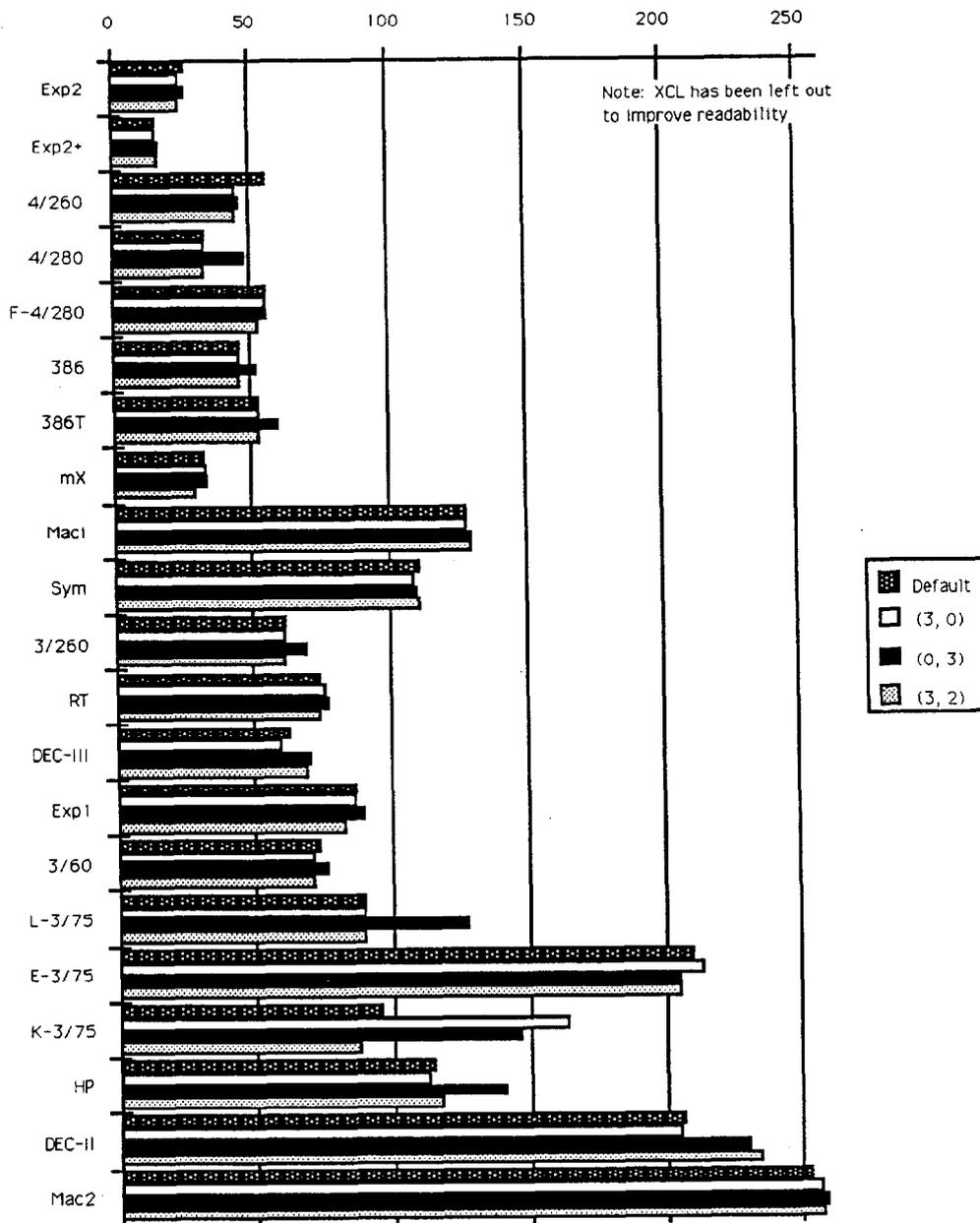


Figure 6: BB1 runs with various OPTIMIZE settings (sec)

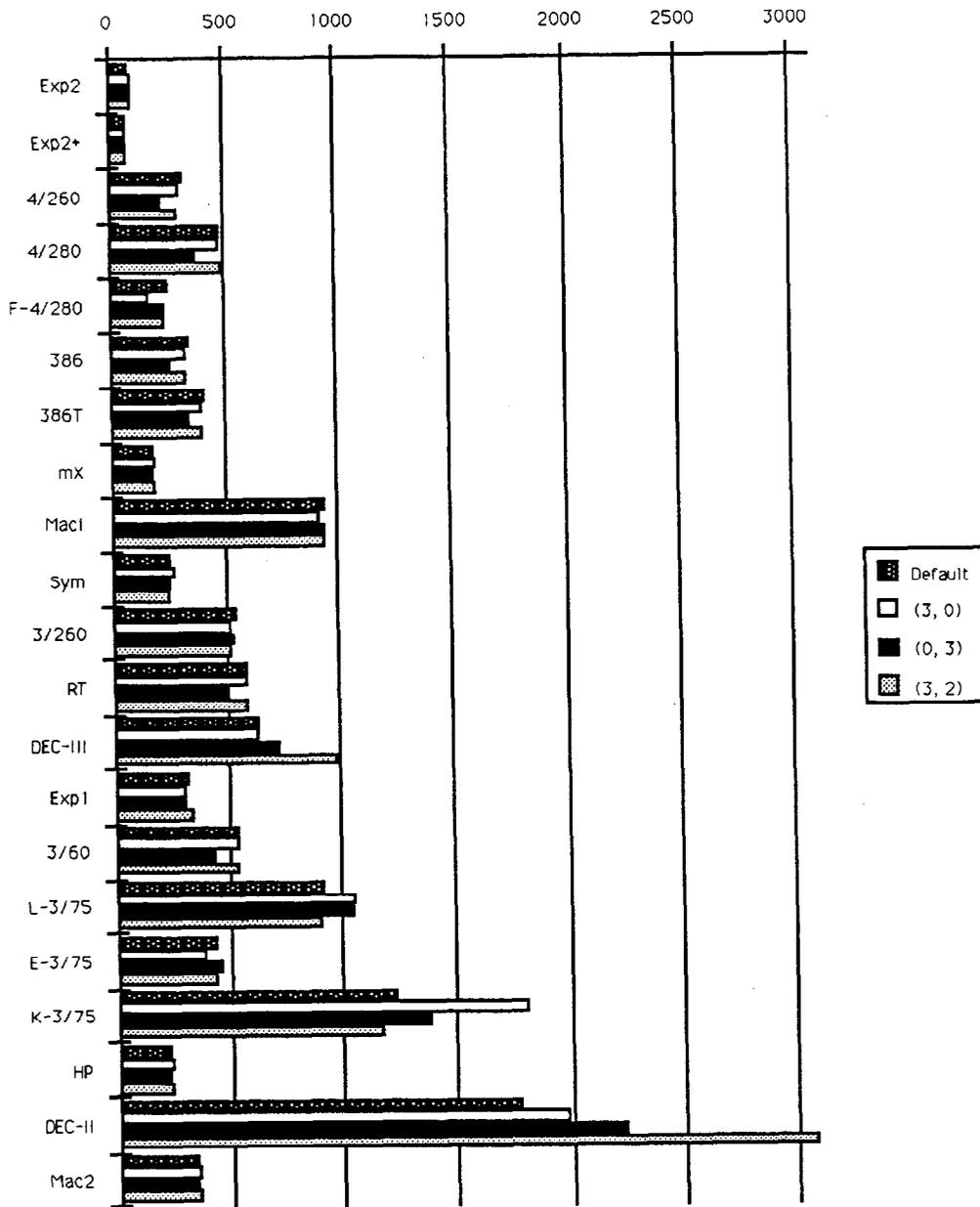


Figure 7: BB1 compilation times with various OPTIMIZE settings (sec)

These charts reveal somewhat surprising results. In several cases, SPEED 3, SAFETY 0 did not give the best results! Lucid Lisp did consistently better when SPEED was higher than SAFETY, as did the HP 9000, and VaxLisp. KCL was definitely behaving strangely with SPEED 0, SAFETY 3 coming out

a good bit faster than SPEED 3, SAFETY 0, with both of those much slower than "default" or SPEED 3, SAFETY 2.

Figure 8 depicts the speedup factor between the slowest time and the fastest time for the BB1 tests with various OPTIMIZE settings.

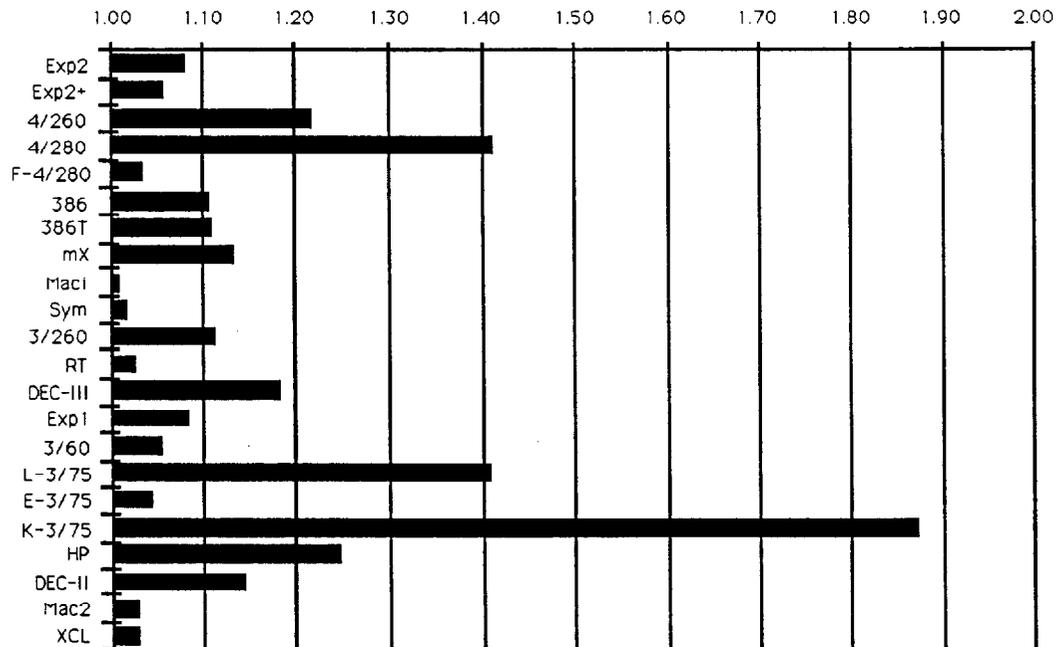


Figure 8: BB1 Speedup Factors Due to OPTIMIZE Settings

8. Effect of Output Reduction on SOAR

The eight-puzzle benchmark for SOAR was originally written when SOAR ran primarily on slower machines than those tested here. Thus it tends to generate a lot of output relative to the amount of computation for some of the modes. For some systems, particularly those with large bit-mapped displays and full-screen windows, this output can be very expensive. To understand the extent of this effect we tested SOAR in the A mode and in the C mode both with full output, and with greatly reduced output (no trace). Table 6 with Figures 9 and 10 show results of these runs. Figure 11 depicts the amount of speedup (ratio of run times) realized by SOAR with reduced output.

Code	Mode A		Mode B	
	Full	Reduced	Full	Reduced
Exp2	33	18	18	16
Exp2+	23	11	13	11
4/260	23	13	11	9
4/280	35	15	14	11
F-4/280	36	36	20	19
386	41	27	21	19
386T	52	31	26	23
mX	50	27	29	27
Mac1	165	65	63	44
Sym	55	40	34	32
3/260	49	33	23	22
RT	61	36	32	28
DEC-III	95	76	95	92
Exp1	90	63	75	71
3/60	66	38	34	31
L-3/75	82	67	45	41
E-3/75	124	109	81	80
K-3/75	186	136	120	111
HP	61	51	52	52
DEC-II	351	283	390	401
XCL	473	390	243	232

Table 6: SOAR Run Times with Full and Reduced Output

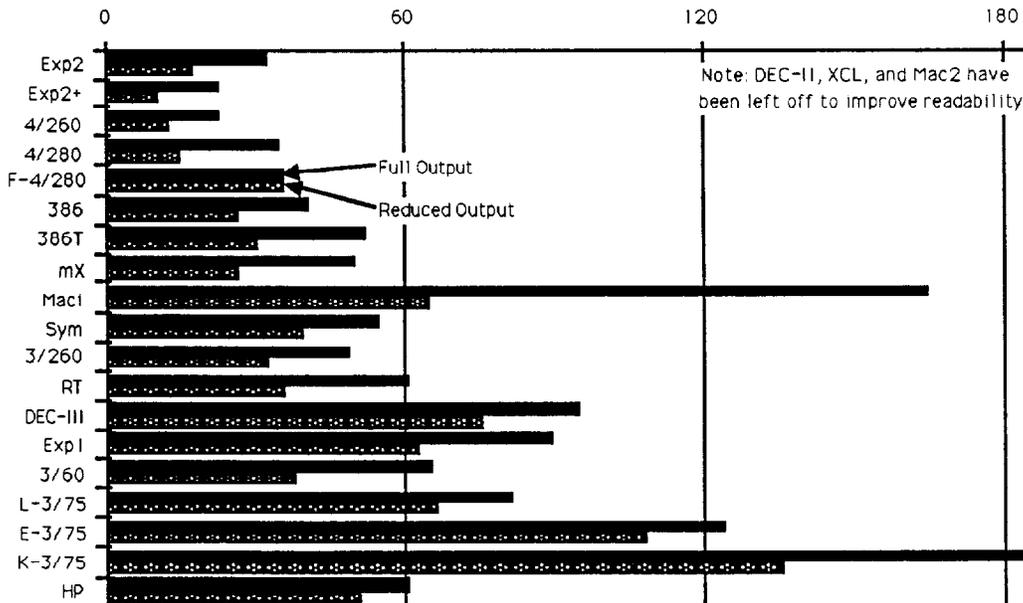


Figure 9: SOAR A Mode (sec)

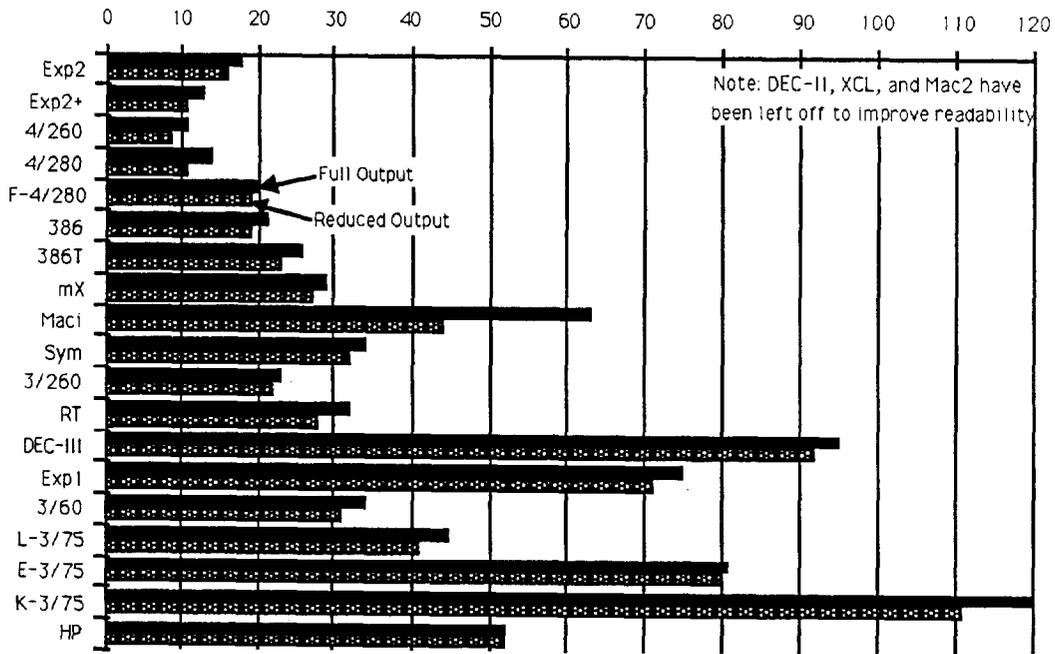


Figure 10: SOAR C Mode (sec)

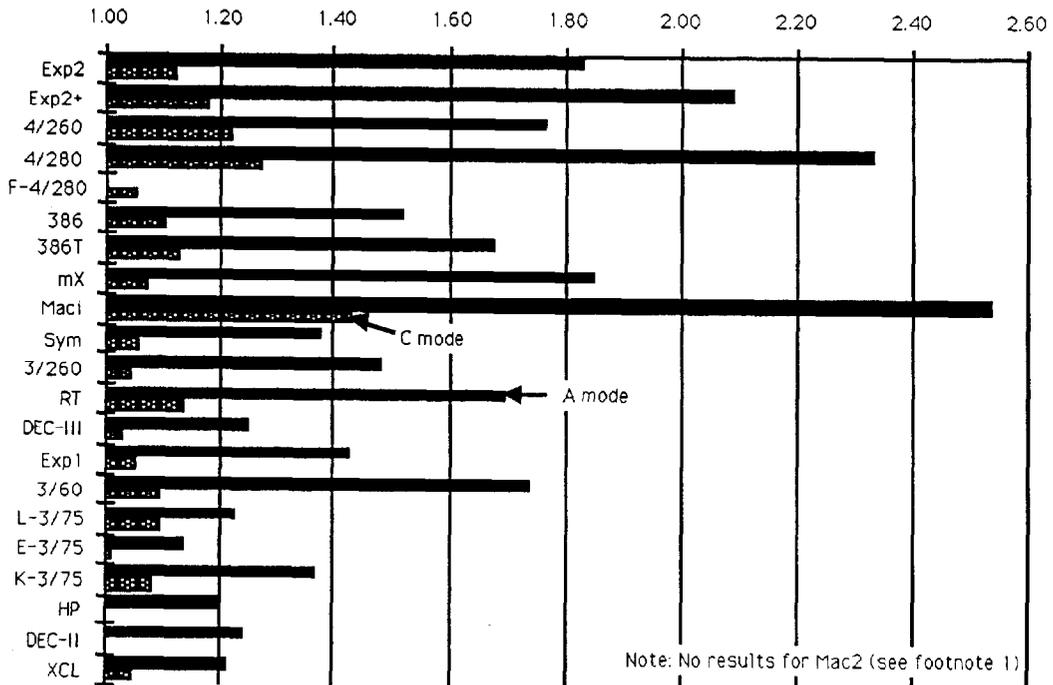


Figure 11: SOAR Speedup Due to Reduced Output

Three factors seemed to influence the speedup with reduced output:

- A fast processor, since the amount of time spent computing versus doing I/O would be reduced, causing a reduction in I/O time to be more significant.
- A larger screen or window since it is expensive to scroll a large area.
- A large-overhead I/O system such as the MacIvory's Dynamic Windows.

9. Future Work

Obvious areas in which this work might be extended include:

- Updating the results to reflect more recent versions of the Common Lisp systems;
- Adding more test systems, especially mainframes;
- Benchmarking other programs besides SOAR and BB1;
- Evaluating the effect of declarations on run times;
- Adding measurements of storage management overhead;
- Collecting more data on I/O overhead;
- Understanding better why platforms vary in performance from application to application and Lisp implementation to Lisp implementation.

10. Conclusions

Two moderate-sized applications, SOAR and BB1, were benchmarked on 22 Common Lisp systems to help in the evaluation of different Common Lisp systems. The run and compile times for these benchmarks were presented and discussed. A large variation was observed between the ranking of systems when running the SOAR test versus the ranking when running the BB1 test. This leads us to conclude that while these experimental results and ones like them can be used to class machines together roughly, it is impossible to use such a set of benchmarks to decide in advance how a given application will perform on a given system. There is no substitute for actually running the program on the systems in question.

Figure 12 shows the average of the normalized¹ run times for the test programs with the systems ranked in order. On the basis of this data, the systems tested may be ranked as follows:

¹ The data were normalized by dividing each by the average of the results for all the tested implementations.

Very Fast (≤ 0.50 anr -- averaged normalized run time): TI Explorer II Plus (Exp2+), TI Explorer II (Exp2), and Sun 4 with Lucid Lisp (4/280 and 4/260)

Fast (> 0.50 anr, ≤ 1.00 anr): TI microExplorer (mX), Compaq 386 (386), Sun 4 with Franz Lisp (F-4/280), Compaq 386 portable (386T), Sun 3/260 (3/260), IBM RT/APC (RT), and Sun 3/60

Medium (> 1.00 anr, ≤ 1.50 anr): Symbolics 3645 (Sym), Sun 3/75 with Lucid Lisp (L-3/75), HP 9000/350 (HP), TI Explorer I (Exp1), and DEC MicroVax III (DEC-III)

Slow (> 1.50 anr, ≤ 2.50 anr): Symbolics MacIvory (Maci), Sun 3/75 with Kyoto Common Lisp (K-3/75), and Sun 3/75 with old Franz Extended Common Lisp (E-3/75)

Very Slow (> 2.50 anr): Apple Macintosh II with Allegro Lisp (Mac2), DEC MicroVax II (DEC-II), and Xerox 1186 (XCL),

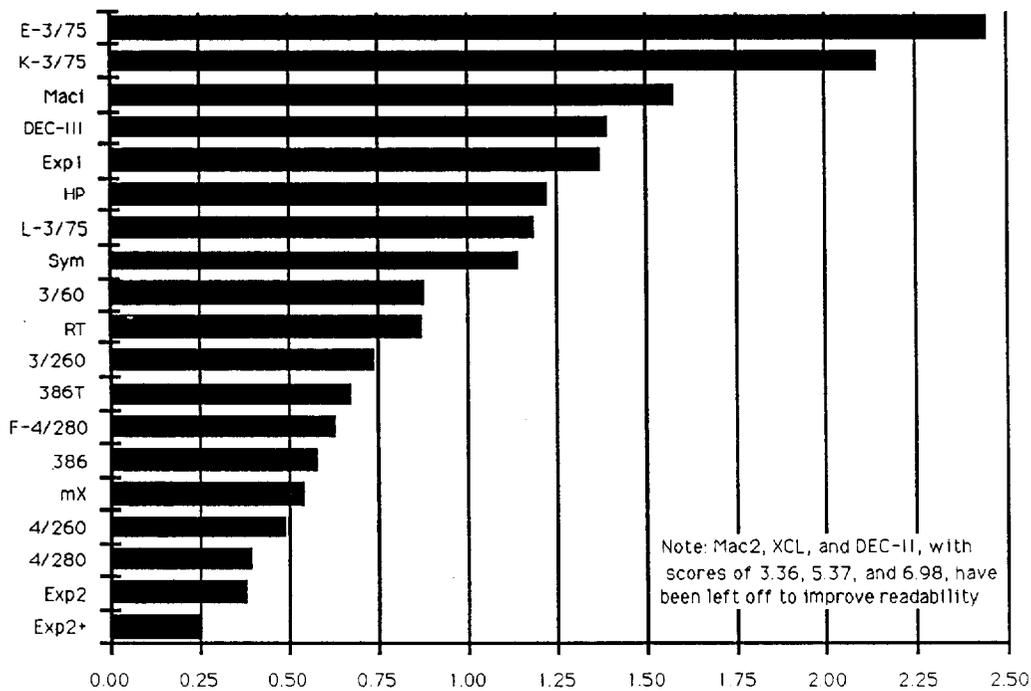


Figure 12: Averaged Normalized Run Times

We were surprised at the high speed of the small 386 machines, and at the slowness of the still early MacIvory, the DEC machines, and the Xerox machine.

Dedicated Lisp machines compile relatively faster than conventional machines, and, generally, conventional machine systems that took more time to compile produced faster code, as one would expect.

While the experiment to measure the effect of different settings of the OPTIMIZE declaration was interesting, with such a small sample no real conclusion about the effect of various OPTIMIZE settings can be drawn. However the indications are that, in the absence of other declarations (e.g. for TYPE), only relatively small gains are available. It is probably best to experiment with various settings to see which gets the best speed for a given program.

Reducing the amount of output that a program generates can have a large effect on the run time of the program, especially when moving the program to a faster machine. This indicates that it is worth taking some time to consider the nature of the I/O system and interaction needed by a program when designing a user interface for a fast-running program.

These results must be used very carefully since they represent only one piece of information about the performance of the very complex systems tested. We have measured only execution speed, but many aspects of the software will impact the development of programs such that in a given amount of time a program might be written for one machine that runs faster and perhaps with fewer errors than a program written in the same amount of time on another machine that ranks faster in these tests due to superior support given to the programmer during development. Do not underestimate the power of the programming environment.

11. Acknowledgements

This work would have been completely impossible without the assistance of many people and companies. Mike Kramer of Texas Instruments Inc. supplied the Explorer II Plus processor board. Eric Warner and Michael Borke of Sun Microsystems Inc. supplied access to the Sun 4 systems and the Sun 3/260 and 3/60 systems. Franz Inc. supplied a test version of Extended Common Lisp. Marty Hollander of Franz Inc. supplied a version of Allegro Common Lisp for the Sun 4. Jeff Harvey of Digital Equipment Corp. arranged access to the MicroVax systems. Susan Rosenbaum and Eric Gilbert of Lucid Inc. supplied access to the Compaq machines and the IBM RT. Bruce Hamilton of Hewlett Packard Inc. arranged access to the HP 9000. Many thanks to all of them.

12. References

- [Gabriel 1985] Gabriel, R. P. **Performance and Evaluation of Lisp Programs**, M.I.T. Press, Cambridge, Massachusetts, 1985.
- [Hayes-Roth 1985] Hayes-Roth, B. *A Blackboard Architecture for Control*, in *Artificial Intelligence Journal*, Volume 26, pp. 251-321, July 1985.
- [Hayes-Roth 1988] Hayes-Roth, B., and Hewett, M. *BB1: An Implementation of the Blackboard Control Architecture*, in **Blackboard Systems**, edited by Robert Englemore and Tony Morgan, Addison-Wesley, 1988, pp. 297-313.

[Laird 1987] Laird, J. E., Newell, A., and Rosenbloom, P. S. *Soar: An Architecture for General Intelligence*, in *Artificial Intelligence Volume 33*, Number 1, pp. 1-64, 1987.

[Steele 1984] Steele, G. L. Jr. **Common Lisp the Language**, Digital Press. 1984

Appendix A -- System Descriptions

This appendix contains detailed descriptions of the systems used in these measurements. In the descriptions, "Code" refers to a short name used to indicate the systems under test. Usually it is the model of the machine except where there is more than one Lisp for a machine (as in the case of the Sun 3/75) in which case a letter is prefixed to indicate the Lisp being used. "Timing Template" indicates how the information reported by the TIME macro was recorded. "Elapsed" indicates the total elapsed time, "run" indicates CPU time used, "gc" indicates time spent in garbage collection, "user" and "system" distinguish between user mode and kernel mode time, and "paging" indicates time waiting for virtual memory disk operations. Code:

Code: **3/260**

Computer Type: **Sun 3/260**

Operating System: **Sun OS 3.4**

Lisp: **Lucid 2.0**

Disk Configuration: **280MB**

Swapping Size: **60MB**

Memory Configuration: **8MB**

Display Configuration: **Color in mono mode**

Other Configuration:

Special Comments: **used :EXPAND 130 :GROWTH-RATE 130**

Timing Template: **elapsed (user-run + system-run)**

Date-of-test: **Summer 1988**

Code: **3/60**

Computer Type: **Sun 3/60**

Operating System: **Sun OS 3.4**

Lisp: **Lucid 2.1**

Disk Configuration: **SCSI 141MB**

Swapping Size: **unknown**

Memory Configuration: **24MB**

Display Configuration: **Hi Res Color in mono mode**

Other Configuration:

Special Comments:

Timing Template: **elapsed (user-run + system-run)**

Date-of-test: **Summer 1988**

Code: **386**

Computer Type: **Compaq 386 (20Mhz 386)**

Operating System: **386/IX 5.3 rev level 1.01 (unix)**

Lisp: **Lucid 2.0**

Disk Configuration: **134MB ESDI**

Swapping Size: **unknown**

Memory Configuration: **10MB; 32kB 20ns cache**

Display Configuration: **terminal**

Other Configuration: **none**

Special Comments: **none**

Timing Template: **elapsed (run)**

Date-of-test: **Spring 1988**

Code: **386T**

Computer Type: **Compaq 386 portable (Toaster)**

Operating System: **386/IX 5.3 rev level 1.01 (unix)**

Lisp: **Lucid 2.0**

Disk Configuration: **40MB**

Swapping Size: **unknown**

Memory Configuration: **10MB; no cache**

Display Configuration: **tiny LCD**

Other Configuration: **tiny display**

Special Comments: **portable version of "386" above**

Timing Template: **elapsed (run)**

Date-of-test: **Spring 1988**

Code: **4/260**

Computer Type: **Sun 4/280**

Operating System: **SunOS 3.2 Gamma**

Lisp: **Lucid 2.1**

Disk Configuration: **unknown**

Swapping Size: **unknown**

Memory Configuration: **32MB**

Display Configuration: **Hi Res color in mono**

Other Configuration:

Special Comments: **used :EXPAND 130 :GROWTH-RATE 130**

Timing Template: **elapsed (user-run + system-run)**

Date-of-test: **Summer 1988**

Code: **4/280**
Computer Type: **Sun 4/280**
Operating System: **SunOS 3.2 Gamma**
Lisp: **Lucid 2.1 beta**
Disk Configuration: **417 (Eagle)**
Swapping Size: **60MB**
Memory Configuration: **8MB**
Display Configuration: **Hi Res mono**
Other Configuration:
Special Comments:
Timing Template: **elapsed (user-run + system-run)**
Date-of-test: **Winter 1988**

Code: **DEC-II**
Computer Type: **DEC MicroVax II/GPX**
Operating System: **VMS**
Lisp: **VaxLisp**
Disk Configuration: **2 x 159MB**
Swapping Size: **3k pg page, 8k pg swap**
Memory Configuration: **16MB**
Display Configuration: **GPX**
Other Configuration:
Special Comments:
Timing Template: **elapsed - gc-elapsed (run - gc-run)**
Date-of-test: **Fall 1987**

Code: **DEC-III**
Computer Type: **DEC MicroVax III (3500)**
Operating System: **VMS**
Lisp: **VaxLisp**
Disk Configuration: **(RD53)**
Swapping Size: **unknown**
Memory Configuration: **16MB**
Display Configuration:
Other Configuration:
Special Comments:
Timing Template: **elapsed - gc-elapsed (run - gc-run)**
Date-of-test: **Fall 1987**

Code: **E-3/75**

Computer Type: **Sun 3/75**

Operating System: **SunOS 3.1**

Lisp: **Franz Extended Common Lisp 2.0**

Disk Configuration: **70MB SCSI**

Swapping Size: **50MB local**

Memory Configuration: **28MB**

Display Configuration: **standard resolution mono**

Other Configuration: **Files on Sun 3/180 NFS server**

Special Comments: **Under suntools**

Timing Template: **elapsed (run + gc)**

Date-of-test: **Fall 1987**

Code: **EXP1**

Computer Type: **Texas Instruments Explorer I**

Operating System: **Explorer Lisp Release 4.1**

Lisp: **Explorer Lisp Release 4.1**

Disk Configuration: **2 x 140MB SCSI**

Swapping Size: **80MB**

Memory Configuration: **8MB**

Display Configuration: **1024 x 768 mono**

Other Configuration:

Special Comments: **TGC (incremental generation scavenging GC) on**

Timing Template: **elapsed - paging**

Date-of-test: **November 1988**

Code: **EXP2**

Computer Type: **Texas Instruments Explorer II**

Operating System: **Explorer Lisp Release 4.1**

Lisp: **Explorer Lisp Release 4.1**

Disk Configuration: **2 x 140MB SCSI**

Swapping Size: **80MB**

Memory Configuration: **16MB**

Display Configuration: **1024 x 768 mono**

Other Configuration:

Special Comments: **TGC (incremental generation scavenging GC)**

Timing Template: **elapsed - paging**

Date-of-test: **November 1988**

Code: **EXP2+**
Computer Type: **Texas Instruments Explorer II Plus**
Operating System: **Explorer Lisp Release 4.1**
Lisp: **Explorer Lisp Release 4.1**
Disk Configuration: **2 x 140MB SCSI**
Swapping Size: **80MB**
Memory Configuration: **16MB**
Display Configuration: **1024 x 768 mono**
Other Configuration:
Special Comments: **TGC (incremental generation scavenging GC)**
Timing Template: **elapsed - paging**
Date-of-test: **November 1988**

Code: **F-4/280**
Computer Type: **Sun 4/280**
Operating System: **SunOS 4.0**
Lisp: **Franz Allegro Common Lisp 3.0.1 beta**
Disk Configuration: **2x900**
Swapping Size: **118MB**
Memory Configuration: **32MB**
Display Configuration: **Hi Res mono**
Other Configuration: **Not running under GnuEmacs; no multi-processing; 4MB initial memory**
Special Comments:
Timing Template: **elapsed (user-run + system-run)**
Date-of-test: **Winter 1989**

Code: **HP**
Computer Type: **Hewlett Packard 9000/350**
Operating System: **Unix**
Lisp: **HP Lisp 1.0**
Disk Configuration: **130MB (7958)**
Swapping Size: **unknown**
Memory Configuration: **16MB**
Display Configuration: **color**
Other Configuration: **under gnuemacs**
Special Comments:
Timing Template: **elapsed - run**
Date-of-test: **Fall 1987**

Code: **K-3/75**
Computer Type: **Sun 3/75**
Operating System: **SunOS 3.1**
Lisp: **Kyoto Common Lisp "September 16, 1986"**
Disk Configuration: **70MB SCSI**
Swapping Size: **50MB local**
Memory Configuration: **28MB**
Display Configuration: **standard resolution mono**
Other Configuration: **Files on Sun 3/180 NFS server**
Special Comments: **Under suntools**
Timing Template: **elapsed - run**
Date-of-test: **Fall 1987**

Code: **L-3/75**
Computer Type: **Sun 3/75**
Operating System: **SunOS 3.1**
Lisp: **Lucid 2.0**
Disk Configuration: **70MB SCSI**
Swapping Size: **50MB local**
Memory Configuration: **28MB**
Display Configuration: **standard resolution mono**
Other Configuration: **Files on Sun 3/180 NFS server**
Special Comments: **used :EXPAND 90 :GROWTH-RATE 90**
Timing Template: **elapsed (user-run + system-run)**
Date-of-test: **Fall 1987**

Code: **Mac2**
Computer Type: **Apple Macintosh II**
Operating System: **Mac OS 5**
Lisp: **Allegro Common Lisp 1.1**
Disk Configuration: **100MB internal**
Swapping Size: **n/a**
Memory Configuration: **5MB**
Display Configuration: **E-machines Big Picture 17" mono**
Other Configuration:
Special Comments:
Timing Template: **elapsed - paging**
Date-of-test: **Spring 1988**

Code: Maci**Computer Type: Symbolics MacIvory****Operating System: Genera 7.3i****Lisp: Genera 7.3i****Disk Configuration: 300MB external****Swapping Size: 25,000kW (122MB)****Memory Configuration: 2,688kW (13MB); 2MB Mac II****Display Configuration: Radius****Other Configuration: Apple EtherTalk****Special Comments:****Timing Template: elapsed - paging****Date-of-test: December 1988****Code: mX****Computer Type: Texas Instruments microExplorer****Operating System: Explorer Lisp 5.0****Lisp: Explorer Lisp 5.0****Disk Configuration: 100MB Rodime****Swapping Size: 60MB****Memory Configuration: 12MB mX processor; 2MB Mac II****Display Configuration: 24" (1280 x 960) Moniterm Viking II****Other Configuration: Apple EtherTalk****Special Comments:****Timing Template: elapsed - paging****Date-of-test: December 1988****Code: RT****Computer Type: IBM RT/APC****Operating System: AIX 2.1.2 (unix)****Lisp: 2.0.5 (Lucid 1.01)****Disk Configuration: "Fast" EESDI controller; 3 x 70MB****Swapping Size: 80k x 512kB blocks (40,960MB)****Memory Configuration: 16MB of "fast" memory****Display Configuration: Moniterm 1024 x 768 mono****Other Configuration: AFT floating point unit; GSL windows****Special Comments: Used :EXPAND 69 to get 6MB semispace;****This should be the fastest RT version now available****Timing Template: elapsed (user-run + system-run)****Date-of-test: Spring 1988**

Code: **Sym**
Computer Type: **Symbolics 3645**
Operating System: **Symbolics Release 6.1**
Lisp: **Symbolics Release 6.1**
Disk Configuration: **368MB**
Swapping Size: **200MB**
Memory Configuration: **8MB**
Display Configuration:
Other Configuration: **FPA, no color**
Special Comments: **EGC on; preliminary indications are that 6.1 performs better in these test than 7.2**
Timing Template: **elapsed - paging**
Date-of-test: **Winter 1988**

Code: **XCL**
Computer Type: **Xerox 1186**
Operating System: **Xerox Lisp, Lyric release**
Lisp: **Xerox Lisp, Lyric release**
Disk Configuration: **40MB**
Swapping Size: **16MB**
Memory Configuration: **3.5MB**
Display Configuration: **19" mono**
Other Configuration:
Special Comments:
Timing Template: **elapsed - gc - paging**
Date-of-test: **Winter 1988**

Appendix B -- Test Procedures

To run a BB1 test the following procedure was followed:

1. The .LISP files were copied to the host under test.
2. Lisp was restarted and any necessary configuration, such as disabling end-of-screen processing, was done.
3. If necessary, a (PROCLAIM '(OPTIMIZE (SPEED X) (SAFETY Y))) form was entered.
4. The form (TIME (LOAD "COMPILE-BB1.LISP")) was entered and allowed to complete without interruption, and the resulting information was recorded. A side effect of loading the COMPILE-BB1 file is that the source files are compiled and loaded.
5. The form (TIME (BB1::TEST-BBEDIT)) was entered and allowed to complete. The results were recorded.
6. Step 5 was repeated, which usually resulted in a better time.
7. Steps 2 through 6 were done a total of 4 times; once skipping step 3 and then for $(X, Y) = (3, 0), (0, 3),$ and $(3, 2)$.

To run a SOAR test the following procedure was followed:

1. The .LISP and .SOAR files were copied to the host under test.
2. Lisp was restarted and any necessary configuration, such as disabling end-of-screen processing, was done.
3. The form (TIME (COMPILE-FILE "SOAR.LISP")) was entered and allowed to complete. The results were recorded.
4. Step 2 was repeated.
5. The following forms were entered:


```
(LOAD "SOAR")
(LOAD "DEFAULT.SOAR")
(LOAD "EIGHT.SOAR")
```
6. The form (TIME (RUN-TASK)) was entered. "1<Return>3<Return>" was immediately typed ahead as responses to the prompts soon to follow. The timing was allowed to complete and the results recorded for the "A mode" test.
7. The form (INIT-SOAR) was entered.
8. Step 6 was repeated with "1<Return>1<Return>" in place of "1<Return>3<Return>" for the "B mode" test.
9. Steps 7 and 8 were repeated with "3<Return>3<Return>" in place of "1<Return>3<Return>" for the "C mode" test.
10. The following forms were entered:

```
(EXCISE EIGHT*MONITOR-STATE)  
(WATCH -1)
```

11. Step 9 was repeated for the "C no trace mode" test.
12. Steps 2 and 5 were repeated to reload SOAR.
13. Steps 7 and 8 were repeated for the "A no trace mode" test.

Appendix C: AIM Management Committee Membership

Following are the current membership lists of the various SUMEX-AIM management committees:

AIM Executive Committee:

SHORTLIFFE, Edward H., M.D., Ph.D. (Chairman)
Principal Investigator - SUMEX
Medical School Office Building, Rm. X271
Stanford University Medical Center
Stanford, California 94305
(415) 723-6970

FEIGENBAUM, Edward A., Ph.D.
Co-Principal Investigator - SUMEX
Heuristic Programming Project
Department of Computer Science
701 Welch Road, Building C
Stanford University
Stanford, California 94305
(415) 723-4879

KULIKOWSKI, Casimir, Ph.D.
Department of Computer Science
Rutgers University
New Brunswick, New Jersey 08903
(201) 932-2006

LEDERBERG, Joshua, Ph.D.
President
The Rockefeller University
1230 York Avenue
New York, New York 10021
(212) 570-8080, 570-8000

LINDBERG, Donald A.B., M.D. (Past Adv Group Chrmn)
Director, National Library of Medicine
8600 Rockville Pike
Bethesda, Maryland 20814
(301)496-6221

MYERS, Jack D., M.D.
School of Medicine
Scaife Hall, 1291
University of Pittsburgh
Pittsburgh, Pennsylvania 15261
(412) 648-9933

AIM Advisory Group:

MYERS, Jack D., M.D. (Chairman)
School of Medicine
Scaife Hall, 1291
University of Pittsburgh
Pittsburgh, Pennsylvania 15261
(412) 648-9933

AMAREL, Saul, Ph.D.
Department of Computer Science
Rutgers University
New Brunswick, New Jersey 08903
(201) 932-3546

COULTER, Charles L., Ph.D. (Exec. Secretary)
Bldg 31, Room 5B41
Biomedical Research Technology Program
National Institutes of Health
9000 Rockville Pike
Bethesda, Maryland 20892
(301) 496-5411

FEIGENBAUM, Edward A., Ph.D. (Ex-officio)
Co-Principal Investigator - SUMEX
Heuristic Programming Project
Department of Computer Science
701 Welch Road, Building C
Stanford University
Palo Alto, California 94305
(415) 723-4879

KULIKOWSKI, Casimir, Ph.D.
Department of Computer Science
Hill Center Busch Campus
Rutgers University
New Brunswick, New Jersey 08903
(201) 932-2006

LEDERBERG, Joshua, Ph.D.
President
The Rockefeller University
1230 York Avenue
New York, New York 10021
(212) 570-8080, 570-8000

LINDBERG, Donald A.B., M.D.
Director, National Library of Medicine
Building 38, Rm. 2E-17B
8600 Rockville Pike
Bethesda, Maryland 20814
(301) 496-6221

MINSKY, Marvin, Ph.D.
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
545 Technology Square
Cambridge, Massachusetts 02139
(617) 253-5864

MOHLER, William C., M.D.
Associate Director
Division of Computer Research and Technology
National Institutes of Health
Building 12A, Room 3033
9000 Rockville Pike
Bethesda, Maryland 20892
(301) 496-1168

PAUKER, Stephen G., M.D.
Department of Medicine - Cardiology
Tufts New England Medical Center Hospital
171 Harrison Avenue
Boston, Massachusetts 02111
(617) 956-5910

SHORTLIFFE, Edward H., M.D., Ph.D. (Ex-officio)
Principal Investigator - SUMEX
Medical School Office Building, Rm. X271
Stanford University Medical Center
Stanford, California 94305
(415) 723-6979

SIMON, Herbert A., Ph.D.
Department of Psychology
Baker Hall, 339
Carnegie-Mellon University
Schenley Park
Pittsburgh, Pennsylvania 15213
(412) 578-2787, 578-2000

Stanford Community Advisory Committee:

SHORTLIFFE, Edward H., M.D., Ph.D. (Chairman)
Principal Investigator - SUMEX
Medical School Office Building, Rm. X271
Stanford University Medical Center
Stanford, California 94305
(415) 723-6979

FEIGENBAUM, Edward A., Ph.D.
Heuristic Programming Project
Department of Computer Science
Margaret Jacks Hall
Stanford University
Stanford, California 94305
(415) 723-4879

LEVINTHAL, Elliott C., Ph.D.
Departments of Mechanical and Electrical Engineering
Building 530
Stanford University
Stanford, California 94305
(415) 723-9037